Diseño e Implementación de una Carga Electrónica Inteligente.

Robert Exneider Velasco Cáceres

Erika Nathalia Zárate Torres

Presentado para optar por el título de Ingeniero Electricista

Director:

Jaime Guillermo Barrero Pérez

Ingeniero Electricista

Universidad Industrial de Santander

Facultad De Ingenierías Fisicomecánicas

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones

Bucaramanga

2019

Agradecimientos

A nuestro director de proyecto de grado Msc Jaime Guillermo Barrero, por convertirse en nuestro mentor e instruirnos no solo con conocimientos académicos si no también con verdaderas lecciones de vida.

A la universidad Industrial de Santander, por permitirnos enfrentar este gran reto y forjar nuestro carácter de la mejor manera posible, con alegrías y tristezas pero siempre con la confianza de que pasaríamos cada una de las metas propuestas para alcanzar nuestro objetivo final.

A cada uno de nuestros maestros porque también son participes de lo que somos ahora y de los que seremos en unos años.

Dedicatorias

Agradezco en primera instancia a Dios, quién supo guiarme, iluminarme y darme las fuerzas suficientes para no desfallecer y hacerle frente a las dificultades que se presentaron en este duro camino.

A mi madre Olga y mi padre Manuel, quienes con su inmenso amor y apoyo me dieron esta gran oportunidad de formación. A la vez de dejarme grandes enseñanzas, hacerme ver la vida de diferente manera y acompañarme en cada uno de los duros momentos siempre deseando y anhelando lo mejor para mí.

A los profesores que me ayudaron todo este tiempo con sus conocimientos, consejos y enseñanzas, en especial a esos que me tuvieron paciencia para guiarme en el desarrollo de todos mis conocimientos.

Por último a mis compañeros de clase, quienes con su apoyo y compañerismo, aportaron en un alto porcentaje la consecución de esta meta, en especial a Erika Zárate, quién ha sido mi mayor apoyo y con su paciencia, conocimiento, motivación y entendimiento me ha colaborado para que este sueño se haga realidad.

Robert Exneider Velasco Caceres

A Dios porque esto es gracia de él. A mis padres porque sin su apoyo esto no sería posible, a mi familia porque son mi bastón, a mis amigos porque también esto es de ustedes.

A mi madre porque sin importar las razones siempre ha estado ahí, a mi padre porque sin reparo contribuyó en esta etapa de mi vida, a ellos dos por ser el trampolín para cumplir este nuevo logro, se merecen este y muchos más, no hay manera de pagarles lo que han hecho por mí.

A mis amigos, nos hicimos el camino más ameno.

A la selección de futbol femenino UIS, ¡qué sería de mí sin ustedes!

A Robert Velasco porque a pesar de las dificultades nada nos quedó grande, hoy alcanzamos esta meta.

¡Gracias a todos, estoy segura que la vida nos reunirá de nuevo!

Erika Nathalia Z

Tabla de Contenido

Pág.

Lista de Tablas

Pág.

Lista de Figuras

Pág.

Lista de Apendices

Pág.

RESUMEN

TÍTULO: DISEÑO E IMPLEMENTACIÓN DE UNA CARGA ELECTRÓNICA INTELIGENTE.

AUTORES: ROBERT EXNEIDER VELASCO CACERES, ERIKA NATHALIA ZÁRATE TORRES

PALABRAS CLAVES: Uso racional de la energía, Energías renovables, baterías, pantalla, ESP32, Arduino, Nextion, ITEAD, Transistor, Microcontrolador.

DESCRIPCIÓN: La creciente demanda energética debido a la necesidad de las personas por llevar una vida más confortable y al crecimiento económico de los diferentes países, ha significado un aumento exponencial de la generación de energía año tras año, de la misma manera que los gases de efecto invernadero han ocasionado daños irreversibles que repercuten en el calentamiento global. Como alternativa a este problema surgen las energías renovables y el avance en otras fuentes no convencionales de energía, las cuales para un mejor aprovechamiento energético requieren de sistemas de almacenamiento (baterías).

El presente trabajo de grado tuvo como fin realizar el diseño de una carga electrónica que permite determinar la energía almacenada en una batería, el diseño se dividió en las siguientes etapas: una de potencia, una de control basada en un microcontrolador que almacena datos de tensión y corriente de la batería, hasta que esta alcanza un umbral previamente definido y las interfaz gráfica para presentarle la información al usuario.

Se seleccionaron los elementos para cada una de las etapas, seguidamente se simularon las etapas por medio de un software para conocer el comportamiento del circuito frente a posibles alteraciones. Por medio del microcontrolador (ESP32) se procesaron las señales de tensión, corriente y temperatura que serán transmitidas por Wi-Fi a un dispositivo móvil en donde se observará el comportamiento de la corriente, además contará con una interfaz gráfica que permite hacer el control de la corriente de manera amigable. Todo esto con el fin de favorecer un mejor aprovechamiento de la energía almacenada en las baterías.

---

* Trabajo de grado

** Universidad Industrial de Santander. Facultad De Ingenierías Fisicomecánicas . Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director: Jaime Guillermo Barrero Pérez. Ingeniero Electricista

ABSTRACT

TITLE: DESIGN AND IMPLEMENTATION OF A SMART ELECTRONIC LOAD.

AUTHORS: ROBERT EXNEIDER VELASCO CACERES, ERIKA NATHALIA ZÁRATE TORRES.

KEYWORDS: Rational Use of Energy, Renewable Energies, Batteries, Screen, ESP32, Arduino, Nextion, ITEAD, Transistor.

DESCRIPTION: Growing energy demand due to people's need to lead more comfortable lives and the economic growth of different countries, has meant an exponential increase in power generation year after year, in the same way that greenhouse gases have caused irreversible damage that impacts global warming, as an alternative to this problem, renewable energy and advancement in other unconventional energy sources emerge, which for better energy use require storage systems (batteries).

The purpose of this present degree work was to design an electronic charge that allows to find the energy stored in a battery, the design was divided into the following stages: one of power, a microcontroller-based control that stores battery voltage and current data, until it reaches a previously defined thres hold and the graphical interface to present the information to the user.

The elements were selected for each of the stages, the stages were then simulated by software to understand the behavior of the circuit against possible alterations. The voltage, current and temperature signals to be transmitted by Wi-Fi were processed via the microcontroller (ESP32) to a mobile device where the behavior of the current will be observed, it will also have a graphical interface that allows you to control the current in a friendly way. All this in order to promote a better use of the energy stored in the batteries

---

* Degree Paper

** Universidad Industrial de Santander. Facultad De Ingenierías Fisicomecánicas . Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director: Jaime Guillermo Barrero Pérez. Ingeniero Electricista

Introducción

Desde tiempos inmemorables se ha presentado el gusto y la fascinación por la electricidad, varias civilizaciones antiguas se cuestionaron la importancia y la necesidad de almacenar energía de alguna manera. La antigua Mesopotamia y los Egipcios observaron que el bronce era conductor y más adelante Tales de Mileto describió los efectos de la electricidad estática (año 624 a.C), tuvieron que pasar muchos siglos para que el italiano Alessandro Volta construyera la pila que conocemos actualmente[1].

Al día de hoy, los gobernantes y las necesidades de sus países han provocado que la demanda de energía crezca exponencialmente, creando la necesidad de cubrir los requerimientos que permitan abastecer la producción de energía para que las personas tengan una vida más placentera, implementando nuevas tecnologías de generación eléctrica que contribuyen con el cuidado del medio ambiente, dichas tecnologías son renovables, por tanto, no se puede controlar lo que se genera como en el caso de las energías eólica y solar ya que son fuentes intermitentes de energía, sin embargo se puede obtener un mayor aprovechamiento si se utilizan tecnologías de almacenamiento que permitan administrar los recursos obtenidos por éstas en las horas de mayor demanda o incluso si el consumo se da durante la noche.

Las tecnologías de almacenamiento presentan diversas capacidades debido a su uso y a su efecto memoria, razón por la cual es importante corroborar sus parámetros mediante la carga electrónica inteligente, ya que ésta nos permite conocer el valor real de la capacidad de almacenamiento y el estado de la batería lo que permitirá dimensionar de mejor manera los sistemas que la requieran, por ejemplo, los fotovoltaicos.

1. Generalidades

1.1 Objetivo general

Diseñar e implementar un circuito electrónico que se comporte como una carga programable para poder determinar la capacidad de almacenamiento en [Ah] de baterías.

1.2 Objetivos especificos

- Seleccionar un transistor de potencia para la implementación de un circuito electrónico que se comporte como la carga de una batería y que permita manejar tensiones de hasta 36 [V] y corrientes de hasta 4 [A].
- Implementar un circuito de control para operar el transistor como una fuente de corriente programable dentro de límites de tensión previamente definidos.
- Establecer por medio de un sistema electrónico la corriente de la batería en función del tiempo y así estimar su capacidad en [Ah].
- Desarrollar una interfaz amigable que le permita al usuario controlar el comportamiento de la carga y visualizar las diferentes señales eléctricas de la batería en una interfaz gráfica.

2. Marco Teórico

2.1 Carga electrónica inteligente

Una carga electrónica inteligente es un dispositivo que simula una carga resistiva y se puede programar para trabajar con corrientes definidas por el usuario, garantizando que mientras la tensión de la batería este sobre un umbral previamente establecido se realicen medidas de tensión, corriente, potencia y energía [2].

2.2 Funcionamiento de la carga electrónica

Debido a que la carga electrónica es un diseño *open hardware*, se pueden encontrar varios esquemáticos y por lo tanto varios diseños que tienen el mismo funcionamiento.

La carga electrónica básica se compone de un amplificador operacional (Opamp) y un transistor, a su vez tiene una resistencia de sensado la cual es la encargada de medir la corriente con la que se va a descargar la batería o la fuente, ésta corriente se ajusta generalmente con potenciómetros.

La carga electrónica presenta dos etapas: una de control operada por el Opamp y otra de potencia compuesta por transistores, de las cuales se tiene que:

- ✓ **Etapa de control:** en esta etapa el circuito cuenta con un amplificador operacional (Opamp), que se comporta como un seguidor o *buffer*, en esta configuración se eliminan los efectos de cargas importantes en las salidas y se adaptan las impedancias, al conectar un dispositivo con una gran impedancia a otro con impedancia pequeña o viceversa. En este caso, la tensión de salida será igual a la tensión de la entrada y la impedancia de entrada, teórica, sería infinita.[3]

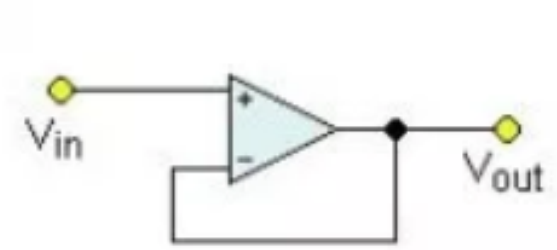


Figura 1. Seguidor [3]

En los diversos esquemas que hay de una carga electrónica, el amplificador operacional hará lo necesario para que los voltajes en su entrada negativa (-) y en su entrada positiva (+) sean iguales (realimentación negativa), lo que quiere decir que va a disminuir o a aumentar el voltaje en su salida para que la tensión en sus dos entradas sean iguales, sin embargo existirá una diferencia de tensión debido al voltaje que el Opamp necesita para generar la tensión de salida, esta diferencia es de 20 a 30 [µV][2]. La tensión de salida del Opamp, es la que se encarga de cambiar el estado del transistor, volviéndolo "más o menos conductor" según lo que se requiera.

- ✓ **Etapa de potencia:** se compone de un transistor, que resulta siendo el elemento que disipe la potencia que sale de la fuente de tensión o de la batería.

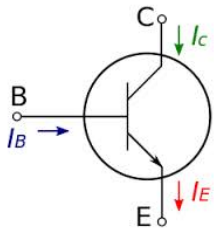


Figura 2. Transistor [4]

La salida del Opamp, se conecta a la base del transistor, en donde su comportamiento dependerá de su configuración (MOSFET o BJT).

- **Funcionamiento de los MOSFET**

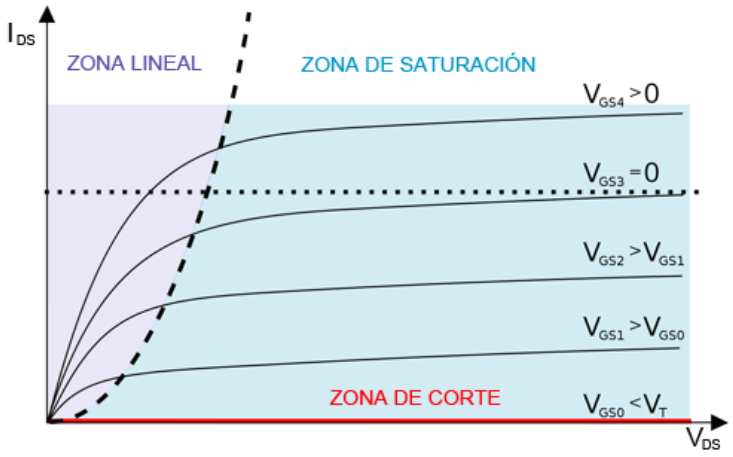


Figura 3. Regiones de trabajo de un MOSFET[5]

El transistor puede trabajar en una de las tres regiones:

- **Corte:** el circuito se encuentra abierto entre dreno y fuente.
- **Saturación:** se comporta como una fuente de corriente controlada por tensión (V_{GS}).
- **Zona lineal:** Se comporta como una resistencia controlada por tensión (V_{GS}).

Para que la carga trabaje, se hace importante que el transistor se comporte como un fuente de corriente, trabajando en saturación, evitando la zona lineal[5]. Para que el transistor opere en la zona de saturación, el voltaje entre el dreno y el surtidor debe ser mayor que el voltaje entre la puerta y el surtidor menos el voltaje de umbral, es decir, $V_{DS} \geq V_{GS} - V_T$. Donde V_T es conocido como el voltaje de umbral, que viene siendo el voltaje de puerta con el cual se produce el canal conductor entre el dreno y surtidor[6]. El valor de la corriente que pasa por el MOSFET es prácticamente independiente de la tensión que se aplica entre el drenador y la fuente, cuanto mayor sea la tensión que se le aplica a la puerta del MOSFET(V_{GS}) , mayor será la corriente que pase por éste[2].

- **Funcionamiento de los BJT**

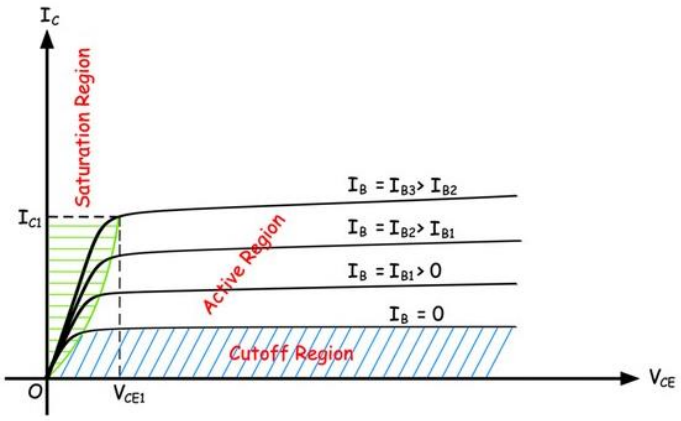


Figura 4. Regiones de trabajo de un BJT[7]

En los BJT las zonas de trabajo varían y se comportan de la siguiente manera:

- **Corte:** Las corrientes que atraviesan el transistor son nulas ($I_{CE} = 0$).
- **Saturación: S**e comporta como una resistencia controlada por corriente (I_B).
- **Activa directa:** es la región entre las zonas de corte y saturación, aquí el transistor se emplea como una fuente de corriente.

En esta ocasión los BJT también se utilizan como fuentes de corriente y deben cumplir con el mismo funcionamiento de los MOSFET operando en la región activa. Para que el transistor opere en la región de saturación la tensión entre el colector y emisor debe estar por debajo de la tensión de umbral V_{CESAT}, lo que conlleva a que la corriente en el colector sea máxima o presente un valor cercano a ella. En un transistor de unión bipolar se controla la resistencia que hay entre el colector y el emisor variando la corriente de la base.

Sin importar el tipo de transistor, la tensión de la batería o de la fuente se queda entre el dreno y el surtidor o el colector y el emisor, siempre que la resistencia de sensado sea muy pequeña, por lo tanto la caída de tensión que se produce en ésta también, ocasionando que la mayor parte

de la tensión se quede en el transistor, siendo éste el que disipe la potencia que sale de la fuente, razón por la cual se le debe agregar un disipador [2].

Generalmente los diseños de una carga electrónica inteligente cuentan con un potenciómetro P1, ver figura 5, conectado al pin positivo del Opamp, con el fin de "igualar" la tensión entre el pin positivo y en el pin negativo (realimentación negativa), permitiendo que la corriente que pasa por la resistencia de sensado sea definida por tensión en el pin no inversor. En donde la corriente viene dada por la siguiente ecuación:

$$I_{out} = \frac{V_{pot}}{R_1} \quad \text{si: } R_1 = 1[\Omega] \text{ entonces } V_{pot} = I_{out} \qquad (1)$$

La corriente que pasa por la resistencia de sensado, es la corriente que queremos definir en la fuente de tensión, panel solar o batería [2].

2.3 Diseño de cargas electronicas

Se puede encontrar gran variedad de diseños, se detallara uno de ellos para ver sus componentes y las etapas que lo conforman.

Las etapas que conforman el diseño se representan de la siguiente manera:

- Etapa de control compuesta por el Opamp.
- Etapa de potencia compuesta por el transistor.

- **Diseño 1**

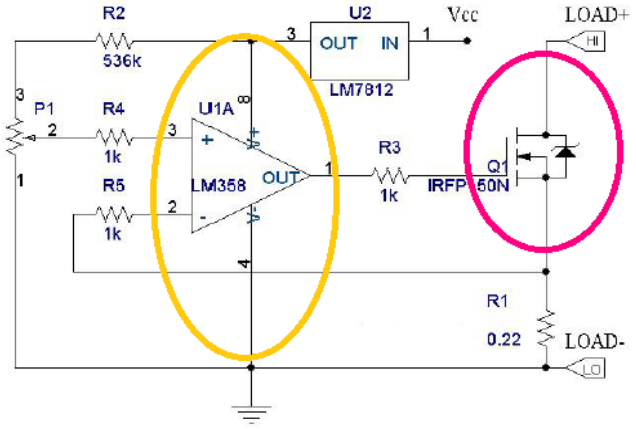


Figura 5. Carga electrónica diseño 1[8]

La figura 5 presenta el diseño de una carga electrónica regulable, se pueden observar las dos etapas (control y potencia), la resistencia de sensado con un valor pequeño(0,22 [Ω]), el potenciómetro que varía la tensión de entrada en el pin positivo del Opamp (V+) ocasionando que la corriente que pasa por la resistencia de sensado y el transistor también varíe, además, posee el LM7812 (regulador de voltaje).

2.4 Baterías

Entre los diferentes elementos activos que podemos probar con la carga electrónica encontramos las baterías, éstas se encargan de convertir energía química en energía eléctrica por medio de reacciones químicas (REDOX) que se generan entre sus placas (placa positiva y placa negativa) y un ácido que funciona como un electrolito para realizar la disolución y correspondiente reacción.

Existen dos grandes grupos en los que se clasifican las baterías:

- **Batería primaría:** después de que la batería se ha descargado, no se puede volver a cargar debido a que su reacción electroquímica es irreversible.
- **Batería secundaría:** después de que la batería se ha descargado, se puede volver a cargar por medio de una fuente externa debido a que su reacción electroquímica es reversible.

De estos se derivan los siguientes tipos de baterías:

✓ **Baterías de plomo-acido:** están constituidas por placas de plomo y un electrolito que por lo general es ácido sulfúrico, son económicas y de fácil fabricación. Son muy comunes en vehículos convencionales, suelen ser de 6[V] y de 12 [V].

Figura 6. Batería plomo-acido[9].

✓ **Baterías níquel-hierro (Ni-Fe):** También conocidas como baterías de ferroníquel, están conformadas por tubos enrollados de láminas de acero niquelado, con un electrolito de hidróxido de potasio contenido en el interior de los tubos, entregando bajos valores de voltaje.

Figura 7. Batería de níquel-hierro[10]

- ✓ **Baterías de níquel-cadmio (Ni-Cd):** Están conformadas por ánodos de cadmio y cátodo de hidróxido de níquel, que a su vez están recubiertos por un electrolito de hidróxido de potasio, éstas al contrario de las baterías de plomo-acido, permiten sobrecargas y/o descargas profundas permitiendo así un mayor número de cargas y descargas, sin embargo presentan el efecto memoria, se usan para juguetes, cámaras y equipos de sonido.

Figura 8. Batería de níquel-cadmio[11]

- ✓ **Baterías de níquel-hidruro metálico (Ni-MH):** Son una extensión de la tecnología níquel cadmio, las baterías de este tipo, emplean un ánodo de hidróxido de níquel y un cátodo de hidruro metálico, son similares a las baterías de níquel-cadmio, diferenciándose entre sí por el menor efecto memoria y mayor capacidad de almacenamiento con el que cuentas las Ni-MH,

pero cuentan con menos ciclos de carga y/o descarga que las Ni-Cd y una afectación mayor a las condiciones climáticas. Han sido las pioneras en la utilización de vehículos eléctricos.

Figura 9. Batería de níquel hidruro metálico[12]

- ✓ **Baterías de iones de litio (Li-Ion):** son los acumuladores más utilizados para pequeños dispositivos electrónicos, están compuestas de un ánodo de grafito y cátodos de óxido de cobalto, trifilina y en algunos casos el óxido de manganeso, son usadas para montajes electrónicos, facilitando así el desarrollo de equipos portátiles e inalámbricos.

Su factor de descarga es bajo, no se ven afectadas por el efecto memoria, presentan gran capacidad de almacenamiento y tienen la facilidad de recibir carga sin completar la descarga total de la batería. Cuentan con un voltaje alto por celda 3.7 [V],

Figura 10. Baterías de iones de litio[13].

- ✓ **Baterías de polímero de litio (LiPo):** Éstas baterías cuentan con el voltaje más alto por celda 3.7 [V], son una variación de las baterías de iones de litio, con mejores características de peso, volumen y autodescarga. Son utilizadas en los dispositivos de la marca Apple y en los carros eléctricos de la marca Hyundai. Al igual que las baterías Li-Ion, tienen un ánodo de grafito y cátodos de óxido de cobalto.

Figura 11. Batería de polímeros de litio[14]

Es importante tener claro algunos conceptos sobre las baterías para comprender mejor la tabla 1 y la tabla 2

- ❖ **Densidad de energía**: es la energía que puede almacenar una batería por unidad de volumen [Wh/volumen][15]
- ❖ **Voltaje:** corresponde a el parámetro básico, el cual determina los usos permitidos para el equipo y porporciona el paso de corriente cuando el circuito se cierra. Cuando la corriente es

muy elevada, la batería tiende a polarizarse ocasionando la disminución en el valor de la tensión.

- **Capacidad de carga:** es la capacidad de electricidad capaz de almacenar la batería y que a su vez puede entregar hasta su completo agotamiento, típicamente se mide en [Ah] o [mAh].
- **Resistencia:** es un valor teórico que se puede obtener mediante los valores de tensión y corriente medidos sobre la batería. Este valor será por consecuencia de las reacciones producidas dentro de la batería, por tanto, este valor varía según avance el tiempo de uso y por ende el desgaste de los electrodos en la batería[15].
- **Estado de carga (SOC):** el SOC es el nivel de energía almacenada dentro de la batería de manera porcentual[15].
- **Profundidad de descarga (DOD por sus sigla en inglés):** es la capacidad de carga capaz de entregar la batería durante su ciclo de trabajo, es inverso al SOC ya que cuando este disminuye el DOD aumenta[15].
- **Máxima corriente de descarga continua:** es la corriente que puede entregar la batería hasta el punto más bajo de carga dado por el fabricante sin que sufra daño alguno[15].
- **Eficiencia:** Es la relación de energía eléctrica entregada por la batería con respecto a la necesaria para cargarla[15].
- ***Delta peak:*** Es un sistema de desconexión de carga que una vez se encuentre la batería a plena carga, desconecta la batería para que no sufra daño alguno, es empleado en baterías NiMH y NiCd[15].
- **Efecto memoria:** manifestación morfológica cambiante de los componentes de la batería con la edad, reduciendo su capacidad por el uso inadecuado debido a descargas y cargas

incompletas. Es decir, la batería podría recordar el nivel de descarga de situaciones anteriores, de modo que al volverse a usar únicamente sea posible descargarla hasta ese nivel[18].

Tabla 1.

Ventajas y desventajas de las baterías

Tipo de Batería	Ventajas	Desventajas
Batería plomo-acido	Suministra picos de corriente altos durante su descarga, posee una tasa baja de autodescarga, alta eficiencia y facilidad para su reciclaje.	No admiten sobrecargas ni descargas profundas, poseen un peso elevado, corta vida cíclica, no aceptan carga rápida, mantenimiento periódico, tienen afectación alta a la corrosión, además son extremadamente contaminantes
Batería níquel-hierro(Ni-Fe)	Son de fácil fabricación, bajo costo, pueden sobrecargarse o descargarse varias veces sin perder su capacidad, no contamina, se componen de abundantes elementos de la corteza terrestre, funciona en rangos de temperatura altos [-40°C,46 °C][19].	Presenta baja eficiencia, tarda horas en cargarse y también en descargarse, además posee baja densidad de energía [19].
Baterías de níquel-cadmio(Ni-Cd)	Presentan una vida cíclica larga, son robustas, admiten sobrecargas y un rango de temperatura muy alto, son fiables.	El Cadmio es un elemento altamente contaminante que exige reciclar las baterías cuando ya no sirven, presentan baja densidad de energía, presentan un precio y un efecto memoria muy elevado.
Baterías níquel hidruro metálico (ni-MH)	Son seguras, presentan un efecto memoria casi despreciable, ofrecen mayor densidad de energía que las baterías de cadmio.	Presentan un nivel alto de auto descarga, vida cíclica media, no pueden ser usadas a bajas temperaturas, es peligrosa si se sobrecarga
Baterías de iones de litio(Li-ion)	Ofrecen gran densidad de energía, cuentan con el voltaje más alto por celda, tienen un factor muy reducido de autodescarga, degradación rápida, casi no se ven afectadas por el efecto memoria.	Sensibilidad a altas temperaturas, rápida degradación, costosas en comparación con las demás baterías.
Baterías de polímeros de litio(LiPo)	Presentan alta densidad de energía, poco peso, poseen la celda con mayor voltaje, no necesitan mantenimiento, bajo factor de autodescarga	Son muy delicadas, ya que se corre el riesgo de deteriorarlas irreversiblemente al descargarlas completamente o sobrecargarlas (4.3 [V]), razón por la cual se debe tener cuidado con los límites de tensión de la batería, requieren circuito de seguridad, precio muy elevado, pueden explotar si se perforan.

La tabla 1 presenta las ventajas y desventajas de las diferentes baterías, con parámetros que son pertinentes a la hora de elegir una de ellas, además en la tabla número 2 se presenta un

cuadro comparativo entre las baterías con parámetros más específicos y que permiten conocer mejor su comportamiento.

Tabla 2.

Comparaciones de las baterías

Tipo de Batería	Energía / peso [Wh /kg]	Tensión de la celda [V]	Vida cíclica (número de recargas)	Tiempo de carga[h]	Autodescarga[%]	Sobrecarga	descarga	Efecto memoria	Capacidad [Ah]
Plomo-acido	30 a 50	2	1000	8 a 16	5	No soporta	No soporta	Medio	7 -960
Níquel hierro (Ni-Fe)	30 a 55	1.2	+10000	4 a 8	10	Soporta	Soporta	-	-
Níquel-Cadmio (Ni-Cd)*	40 a 80	1.25	500	10 a 14	30	soporta	La necesaria	Muy alto	0.5 a 1
Níquel hidruro metálico (Ni-Mh)*	60-120	1.25	1000	2 a 4	20	No se recomienda	Recomendable	Bajo	0.5 a 10
Iones de litio (Li-ion	110 a 160	3.7	4000	2 a 4	10	Soporta	Falla a 2.5 [V]	No tiene	-
Polímeros de litio (Li-Po)	100-130	3.7	5000	1 a 1.5	10	Soporta	Falla a 2.5 [V]	No tiene	-

En la tabla 2:

El número de cargas de las baterías es un valor aproximado.

*Las baterías de níquel se pueden cargar rápidamente, sin embargo acelerar éste proceso puede ocasionar calentamiento en exceso y diminución de su vida útil (cargas rápidas), además son las únicas que admiten este tipo de carga

Las baterías presentan diferentes voltajes de trabajo que se deben tener en cuenta para que su vida útil no se vea afectada: el **voltaje nominal**, la **tensión máxima de carga,** y la **tensión mínima de descarga**.

Tabla 3.

Tensiones de carga y descarga de las celdas de las baterías.

Tensiones	**Plomo-Acido**	**Níquel-Hierro**	**Níquel-Cadmio**	**Níquel-hidruro-metálico**	**Iones de litio**	**Polímeros de litio**
Tensión máxima de carga [V]	2.4	1.7	1.4	1.4	4.2	4.2
Tensión nominal[V]	2	1.2	1.25	1.2	3.7	3.7
Tensión mínima de descarga [V]	1.75	1.1	1	1	2.8	2.8

3. Prototipo

3.1 Diseño del prototipo

El diseño de la carga electrónica inteligente se desarrolló en la Universidad Industrial de Santander con el fin de comprobar el estado de las baterías, para esto se estudiaron los elementos que componen cada una de sus etapas y se eligieron los que presentan el mejor comportamiento para cumplir con los objetivos planteados.

3.1.1 Elementos.

- **Esp32:** se escoge este microcontrolador debido a que permite adquirir y procesar datos de manera rápida y segura, garantizando el funcionamiento correcto de la carga, está compuesto por bloques básicos con el fin de dar y recibir órdenes, a su vez se aprovecha la capacidad criptográfica empleado en su diseño de bajo consumo que integran los módulos Wi-Fi y Bluetooth, con el fin de realizar la conexión digital a internet y mantener los datos en la nube para que sean de fácil acceso para los usuarios desde cualquier lugar en que se encuentre y usando cualquier dispositivo, es decir, el desarrollo de IoT (internet de las cosas), presenta una amplia gama de aplicaciones, el uso del Wi-Fi permite una comunicación a mediano alcance, conectarse a una red LAN y conectarse a internet por medio de un *router*, en cuanto el Bluetooth nos permite conectarnos directamente a otro dispositivo[20].

Presenta un corriente inferior a 5 [uA] en reposo, lo que lo hace ideal para aplicaciones electrónicas portátiles, cuenta con dos núcleos de CPU que se pueden controlar individualmente y la frecuencia del reloj se ajusta entre 80 [MHz] y 240 [MHz]. Además, tiene amplio conjunto

de sensores táctiles capacitivos, sensores de efecto hall, amplificadores de bajo nivel de ruido, interfaz para SD, Ethernet, SPI, UART, I2S e I2C. El modulo trabaja a 3.3[V][20].

Figura 12. ESP32[21]

Para la programación de este módulo se encuentran varios entornos de desarrollo, entre los que encontramos:

- ESP-IDF
- Arduino IDE
- Mongoose IDE
- Javascript Engines
- MicroPython IDE
- LUA-NodeMCU
- Zerynth
- Visual Studio Code

En el código desarrollado se realizó tanto la programación del microcontrolador ESP32, como de la pantalla y de los valores a mostrar en esta misma. Sin embargo a la hora de trabajar con el Wi-Fi, los dos ADC de la tarjeta presentaban inestabilidad en la recepción de los datos, en especial el ADC2, razón por la cual se hizo pertinente el uso de un arduino pro mini para la lectura de datos de los sensores para luego ser enviados a la ESP32.

La programación de la Esp32 se realizó en Visual Studio Code debido a diversos factores, entre los que encontramos la facilidad en cuanto al uso y/o modificaciones a las librerías empleadas, similitud en lenguaje de programación a C++ y/o Arduino IDE, los cuales son lenguajes más comunes de entender e implementar.

- **Arduino pro-mini:** es un dispositivo que sirve para ser empotrado, es potente y ligero.

Figura 13. Arduino pro-mini[22]

La programación del arduino pro-mini se elaboró en Arduino IDE. En este dispositivo se realizó la adquisición de todas las señales eléctricas (tensión, corriente de la batería y temperatura) que luego se enviaron a la ESP32 para el procesamiento de los datos.

- **Nextion:** pantalla táctil que permite realizar la interfaz hombre maquina (HMI) con el fin de facilitar la comunicación de máquinas o equipos con los humanos, comprendiendo de manera fácil cualquier orden o acción que se dé. En este caso, se sabe que la Nextion es un dispositivo de fácil adaptación a cualquier persona ante cualquier situación, ya que sus componentes permiten un ágil procesamiento de datos.

Figura 14. Pantalla Nextion [23]

La pantalla Nextion es una pantalla táctil ideal para visualizar las variables en el momento de realizar la descarga de la batería. A su vez, esta presenta gran variedad de componentes como botones, gráficos, sliders, instrumentos de medida, los cuales permiten enriquecer su interfaz siendo amigable con el usuario. No consume recursos del procesador, desempeñándose de mejor manera que las pantallas LCD que se pueden encontrar en el mercado, su interfaz se desarrolla en el editor de Nextion (Nextion Editor), a la vez que se desarrolla en Visual Studio Code y/o Arduino IDE, los cuales facilitan todas las opciones, comandos y librerías necesarias para un correcto funcionamiento de esta.

- **Amplificador Operacional (OPAMP):** los amplificadores operacionales son muy útiles para modificar los valores de las señales de salida, en este caso, las provenientes de la ESP32, algunos Opamp usados fueron:
  - **Buffer:** es el encargado de mantener los voltajes de entrada (positivo y negativo) a igual valor, además es el que realiza el control de la corriente que fluye de la batería, sin embargo, esta etapa de control no está presente en el prototipo que se plantea por razones que se mencionan más adelante.

- **Amplificador Operacional (inversor y no inversor):** debido a que la ESP32 permite voltajes de hasta 3.3 [V], se hace necesario el uso de dos etapas más: una de amplificación y otra de reducción de la magnitud de la señal que serán procesadas por medio del microcontrolador de la ESP32.

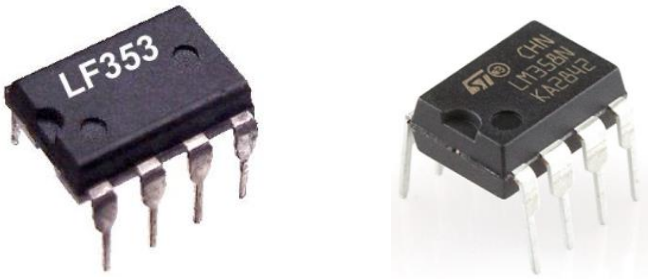


Figura 15. LF353 y Lm358[24][25]

Los dispositivos adecuados para cumplir con estas operaciones son: el LF353 y el LM358, en el caso del LF353 encontramos que presenta un *Slew Rate* alto con un valor de 13 [V/us], además posee una impedancia de entrada elevada ($10^{12}[\Omega]$) ideal para acoplarlo a sensores y a pequeñas señales, dispone de un rango de polarización de ±18 [V] facilitando la conexión, además, es un amplificador operacional de bajo ruido y de bajo consumo. En cuanto al LM358 presenta un *Slew Rate* muy bajo con un valor de 0.5 [V//us], es un operacional con alta ganancia, bajo consumo de potencia y una impedancia de entrada de 10 [M Ω], además posee bajo *offset*. Estos dos amplificadores se eligieron debido a su amplio campo de aplicación principalmente para los sensores.

- **Transistores:** son los encargados de conmutar y amplificar señales, para este prototipo se seleccionaron:

- **IRF3205.**

Figura 16. MOSFET IRF3205[26]

El transistor opera en la zona de saturación y en la de corte, operando como una fuente de corriente que soporta la mayoría de la potencia que entrega la batería. El IRF3205 es el dispositivo electrónico adecuado para cumplir con los propósitos antes mencionados debido a que la corriente máxima que puede pasar por el dreno es de 110 [A], la diferencia de potencial entre el dreno y el surtidor que soporta el dispositivo es de 55[V] y alcanza una potencia de hasta 150[W], superando así los requisitos del diseño de la carga electrónica, además este dispositivo electrónico soporta una temperatura de hasta 175 [°C] haciéndolo ideal para este procedimiento.

- **TIP3055**

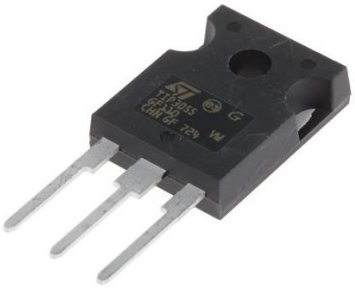


Figura 17. Transistor bipolar TIP3055[27]

El TIP 3055 es un transistor bipolar NPN operado por la corriente que pasa por la base, trabajando en la zona de saturación con una ganancia de corriente $h_{fe} = 70$, y con un voltaje de saturación colector-emisor de 1.1 [V], además este transistor presenta capacidades de corrientes de hasta 7[A] para la base y de hasta 15 [A] para el colector, con una tensión máxima entre colector y emisor de 70[V], por último este dispositivo soporta una temperatura de 150 [°C] cumpliendo con los parámetros necesarios para que sea el dispositivo a emplear para la carga electrónica.

- **ACS712:** Es un sensor que mide corriente en AC o en DC, trabaja con el efecto Hall detectando el campo magnético que se produce por inducción de la corriente con la que se trabaja. Presenta una salida de voltaje proporcional a la corriente, se puede encontrar para tres rangos de corriente de 5[A], 20[A] y hasta 30 [A]

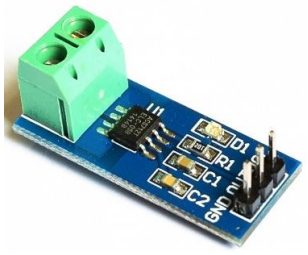


Figura 18. Sensor de efecto Hall[28]

El sensor que se eligió es de hasta 5 [A] debido a que presenta mayor sensibilidad que los demás, cumple con la corriente máxima que debe soportar la carga electrónica y posee una resistencia interna muy pequeña de1.2 [mΩ] permitiendo que la caída de tensión también lo sea y que las pérdidas sean casi despreciables. Es un sensor con alta precisión y con resistencia a picos de corriente de hasta de 5 veces la nominal.

Para una corriente de 0 [A] la tensión que entrega el sensor será $\frac{V_{cc}}{2}$, en donde Vcc tiene un valor entre 4.5-5 [V]. La salida de voltaje y la corriente presentan una relación lineal, dada por la siguiente ecuación:

$$V = m * I + 2.5$$

En donde la m es la sensibilidad y equivale a la pendiente, para conocer la corriente que está pasando por el sensor se debe despejar la corriente de la ecuación anterior

$$I = \frac{V-2.5}{sensibilidad} \quad [28]$$

3.2 Simulaciones

Antes de realizar el montaje del circuito es pertinente elaborar las simulaciones que permiten analizar su comportamiento, para esto se utiliza el software OrCAD como la herramienta que posibilita ver las tensiones y las corrientes en cada uno de los elementos del circuito.

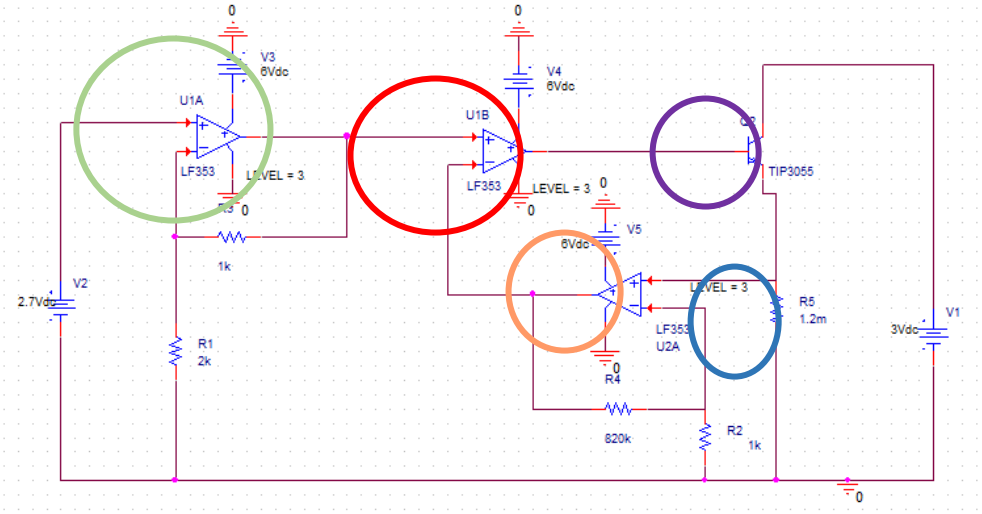


Figura 19. Simulación N° 1 del circuito

La primera simulación que se realizó requiere de una etapa de amplificación debido a que el voltaje máximo de salida de la ESP32 es de 3.3[V], ésta etapa de amplificación tienen una ganancia de 1.5 para que la entrada en el pin positivo del segundo opamp que actúa como *buffer* alcance un valor de 4[V]. El *buffer* es la etapa que realiza el control de la corriente que pasa por la resistencia de sensado, su entrada debe ser de 4[V] para que la corriente con la que se descarga la batería también sea de 4 [A], es decir, que la caída de tensión en la resistencia de sensado es:

$$V_{R5} = 4\ [A] * 1.2\ [m\Omega] = 4.8\ [\text{mV}] \qquad (2)$$

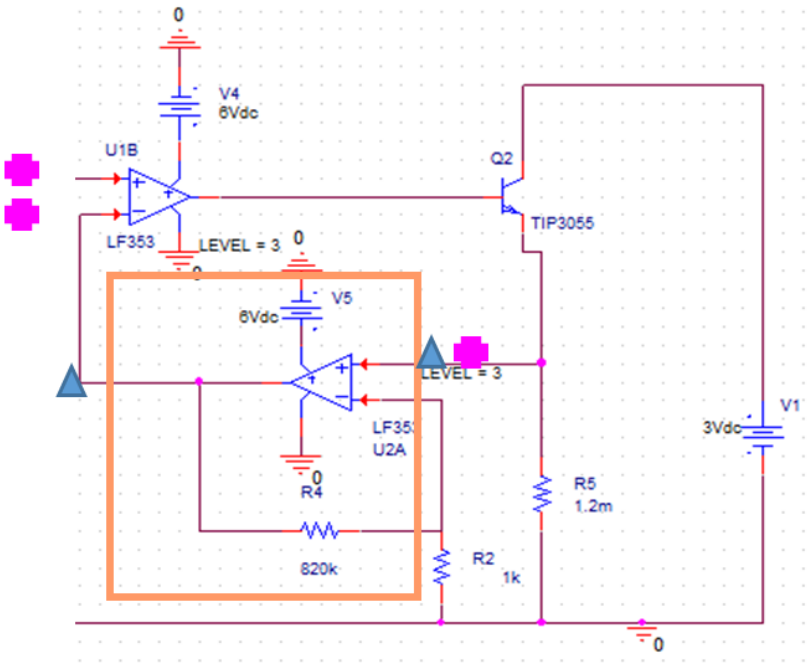


Figura 20. Puntos de tensiones iguales

Se implementó la segunda etapa de amplificación debido a que la caída de tensión en la resistencia de sensado es del orden de los milivolitos y no se perciben las variaciones, con una ganancia correspondiente a:

$$U2A = \frac{4[V]}{4.8[mV]} = 833.33[\text{V/V}] \qquad (3)$$

En la siguiente figura se puede ver la simulación con los valores de tensión y corriente con los que trabajó el circuito.

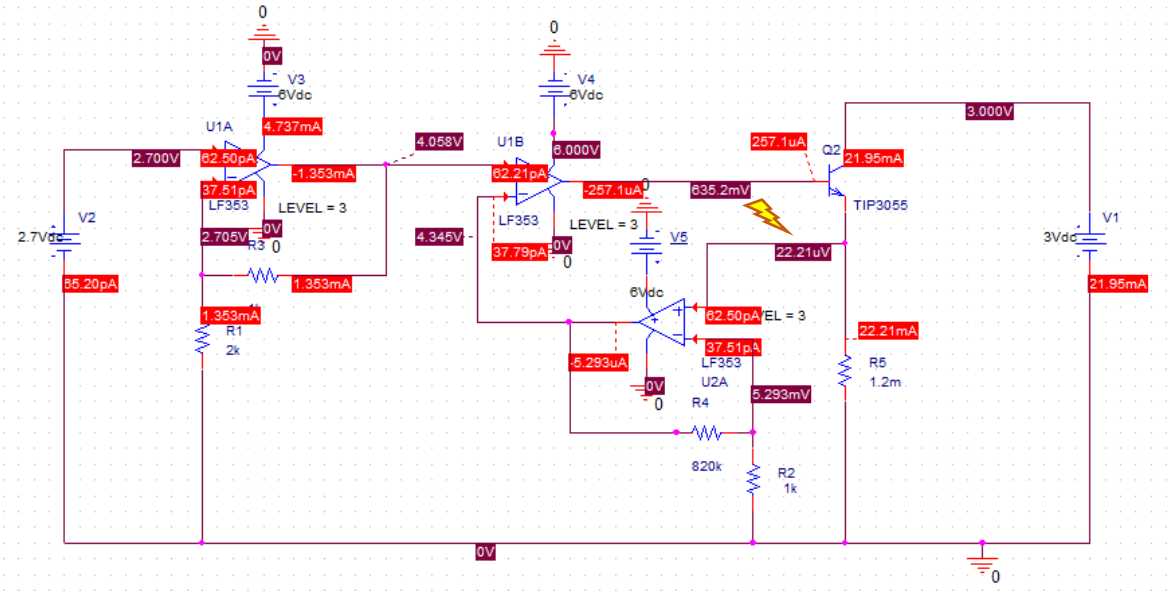


Figura 21. Simulación N°1 con tensiones y corrientes

Debido a que la resistencia del sensor de efecto Hall es muy pequeña 1.2[mΩ], su caída de tensión es de 4.8[mV] cuando pasa la máxima corriente permitida (4[A]), luego su caída de tensión también es muy pequeña.

Como se ve en la figura anterior la primera etapa de amplificación y el buffer están trabajando de manera adecuada pero en la segunda etapa de amplificación el operacional no está trabajando

de manera correcta, ya que las tensiones en los pines de entrada positiva y negativa son diferentes cuando deberían ser iguales, esto ocurre porque el LF353 presenta una tensión de compensación de entrada (*Input Offset Voltage*) de 5[mV] y su tensión de entrada en operación es de microvoltios o alcanza su máximo valor de 4.8 [mV] cuando pasa la corriente máxima sin superar la tensión de offset, debido a esto el operacional no amplifica la señal y no realiza el control de la corriente de la batería. Generalmente la tensión de *offset* no es importante cuando el voltaje de entrada es mayor a 1 [V], pero cuando se trata de las señales de sensores este juega un papel muy importante [29].

Por lo anterior se decidió mejorar el diseño y realizar una nueva simulación del circuito pero esta vez con un transistor MOSFET.

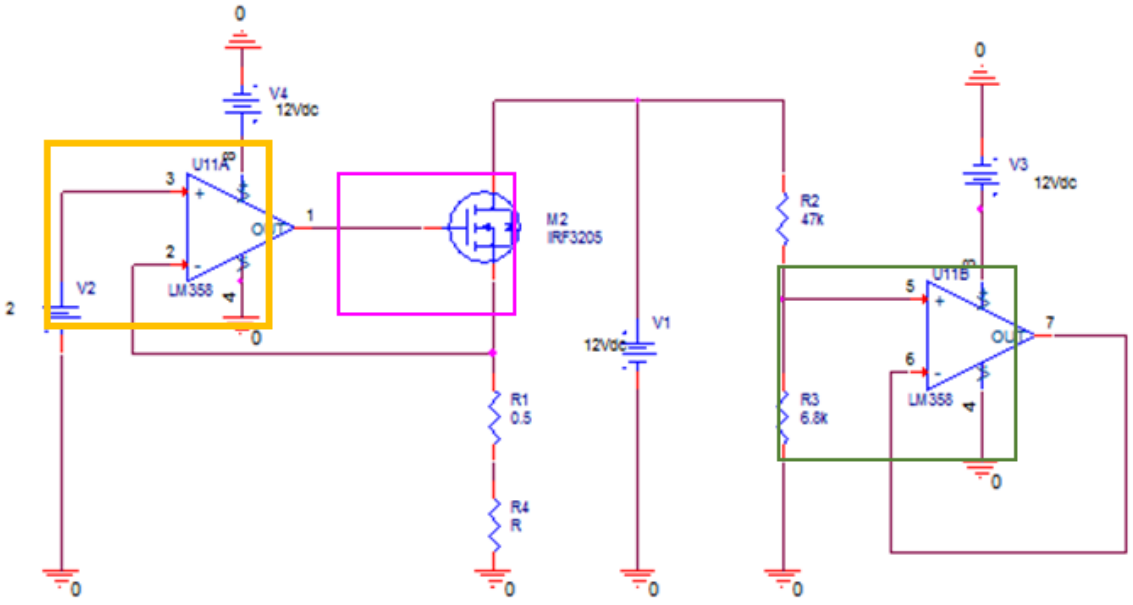


Figura 22. Simulación N° 2 del circuito

La segunda simulación del circuito presenta en su primera etapa un opamp que tiene como función mantener la corriente constante según el voltaje de referencia que se le aplique en su

entrada no inversora (V2), la segunda etapa se compone del transistor (IRF3205) de potencia en serie con una resistencia de 0,5 [Ω] y de la resistencia del sensor de corriente de 1.2 [mΩ] que sirve para medir la corriente con la que se está descargando la batería, R1 se dejo con el fin de mejorar la estabilidad del circuito. En la última etapa aparece un divisor de tensión (R2, R3) y un OPAM que opera como buffer y permite que el uC lea la tensión de la batería de forma segura.

Se eligió esta configuración porque permite controlar grandes corrientes con tensiones pequeñas.

La variación de la tensión de entrada al opamp (V2) con respecto a la corriente con la que se descarga la batería se puede ver la siguiente gráfica, para una batería de 12[V]. Como su funcionamiento fue el esperado, este fue el circuito que se implemento fisicamente.

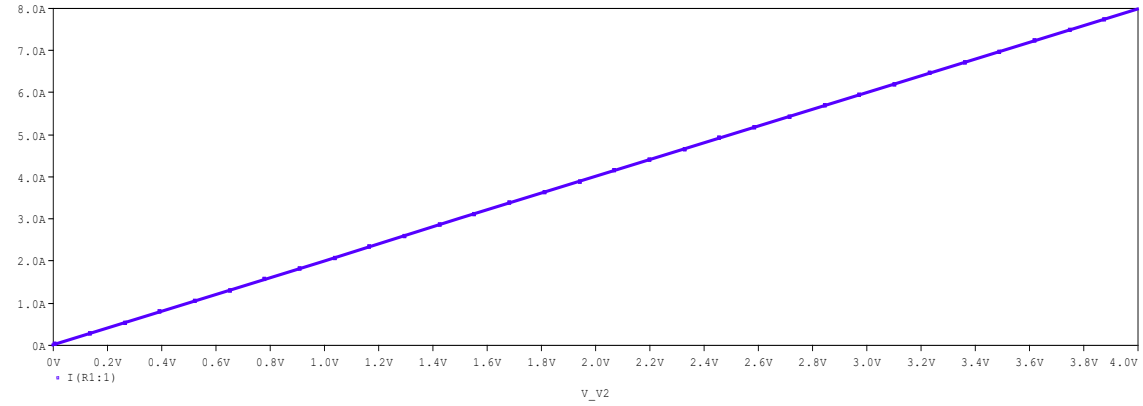


Figura 23. Tensión de entrada vs corriente de descarga batería (12[V])

3.3 Implementación de la carga electrónica

La carga electrónica diseñada soporta una potencia de 144[W], con tensión de hasta 36[V], y corriente de hasta 4[A], el diseño del prototipo se realizó en cinco módulos diferentes:

- ✓ **Nextion:** es el medio de visualización (pantalla) de las variables importantes como temperatura, tensión, corriente, potencia en el lugar donde se encuentra ubicada la carga.

- ✓ **PCB:** en esta placa se realiza el montaje del circuito electrónico, el cual está conformado por la esp32, sensores y la etapa de potencia de la carga electrónica, los cuales se interconectan para brindar una lectura correcta de las distintas variables.
- ✓ **Visual Studio Code:** plataforma en donde se realiza la programación de la ESP32 y la Nextion
- ✓ **Arduino IDE:** plataforma en donde se realiza la programación arduino pro-mini.
- ✓ **Interfaz:** corresponde a la API que permite ingresar de manera remota los parámetros con los que el usuario desea descargar la batería, de igual manera proporciona la visualización gráfica de las distintas señales de la carga que permiten verificar su correcto funcionamiento.

3.3.1 Nextion. La pantalla Nextion es el medio de visualización de las diferentes señales en el lugar donde se encuentra ubicada la carga.

Figura 24. Pantalla de inicio de la Nextion

La pantalla de inicio de la Nextion (Figura 24) permite al usuario relacionarse con el prototipo, de modo que tenga un entorno amigable. Al iniciar el proceso de descarga de la

batería, la pantalla de manera automática muestra el valor en tiempo real de las diferentes variables.

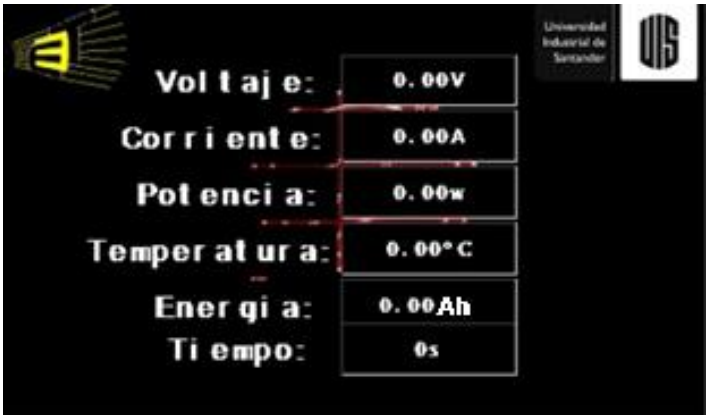


Figura 25. Pantalla de visualización de variables de descarga de la batería.

Las variables se muestran en una página (figura 25) diferente a la página de inicio, estas son:

- **Tensión:** corresponde a la tensión en voltios de la batería a medida que esta se va descargando.
- **Corriente:** es la corriente con la que se está descargando la batería.
- **Potencia:** corresponde a la potencia con la que se está descargando la batería.
- **Temperatura:** Muestra la temperatura del transistor, el cual disipa la potencia entregada por la fuente.
- **Energía:** capacidad de la batería en [Ah], se calculó encontrando la potencia instantánea de la batería (V*I) cada 100[ms] y sumando las áreas, es decir que se determinó el valor de la integral de la potencia:

$$\text{Energía en [J]}=\int_{t_1}^{t_2} p(t)dt \qquad (4)$$

Luego se divide por la tensión nominal de la batería, obteniendo al final el resultado típico de capacidad en [Ah]. Se calcula éste valor hasta que la prueba termine, ya sea por tiempo o porque la batería alcanzó su nivel minimo de tensión,

- **Tiempo:** intervalo de tiempo en el que la batería se descarga.

3.3.2 PCB. La PCB de la carga electrónica fue diseñada en un software denominado EAGLE que permite pasar del esquemático *al protoboard*, y de allí al circuito impreso de manera sencilla y rápida.

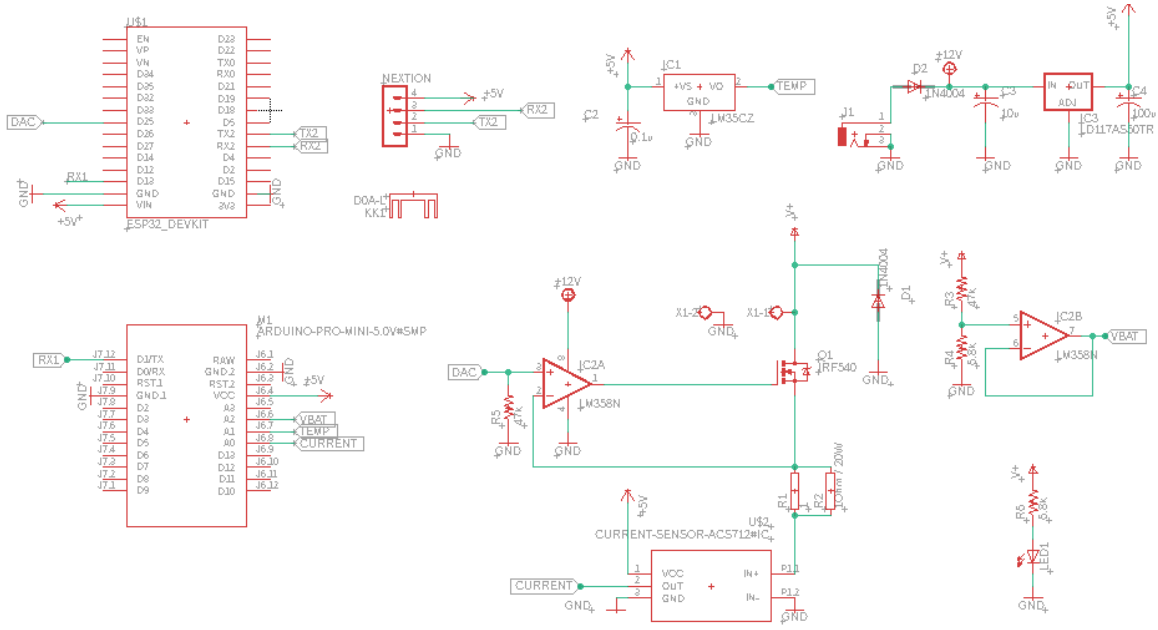


Figura 26. Esquemático de la carga electrónica

Después de realizar el esquemático se construyó el circuito impreso en el programa, para así proceder a trazar los caminos en la PCB finalizando con su construcción. En las siguientes imágenes se pueden observar el resultado final de cada uno.

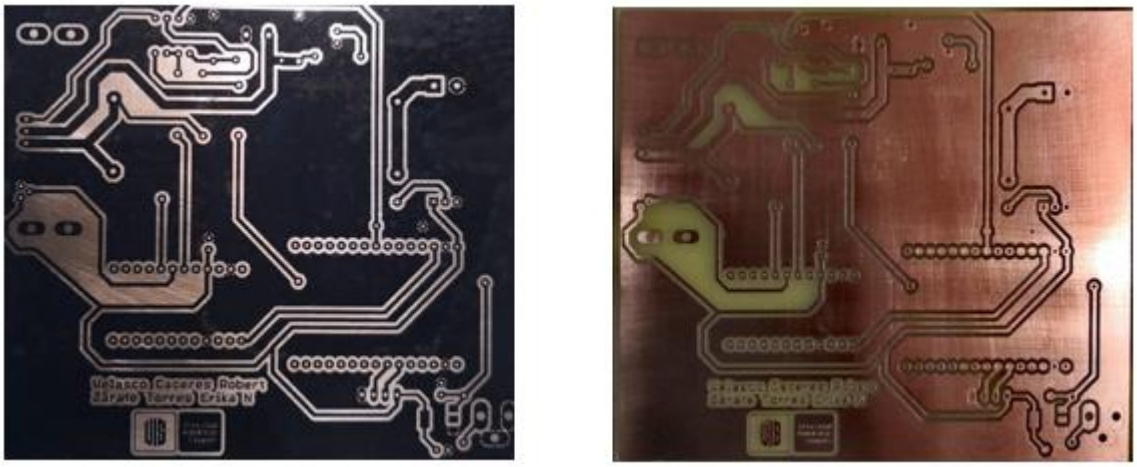


Figura 27. PCB Carga electrónica

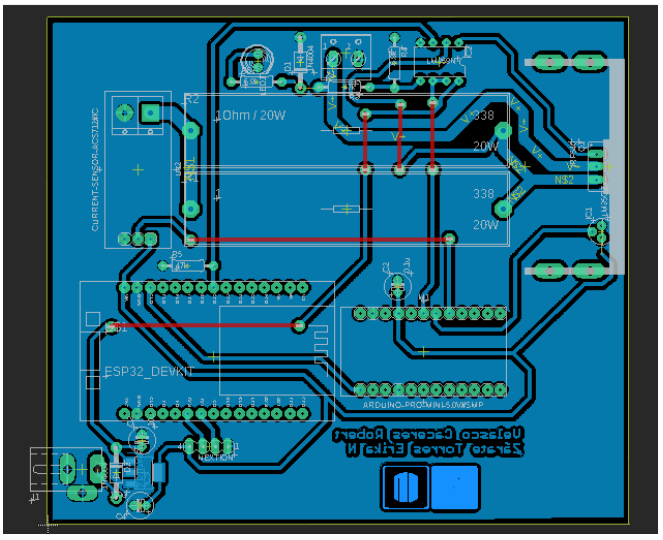


Figura 28. PCB en EAGLE

3.3.3 Visual Studio Code. Se empleó la tarjeta de desarrollo ESP32 por las características que posee, en especial porque es de bajo costo, de fácil acceso y cuenta con distintas plataformas de programación e incluye el módulo Wi-Fi. Este dispositivo está orientado para aplicaciones de IoT, esencial para que el usuario realice el monitoreo remoto de la carga electrónica desde cualquier dispositivo móvil sin importar el lugar en que se encuentre.

La ESP32 realiza el control y el monitoreo de la corriente con la que se descarga la batería, es el dispositivo que permite la comunicación serial con la Nextion, elabora la recepción y el envío de datos por Wi-Fi al servidor de la página web, permite la conexión a la red, limita y asigna los valores a los diferentes pines. Su programación se puede ver en el APENDICE H. Visual Studio Code. Consta de dos archivos denominados config y main, en donde uno permite la configuración de los parámetros y el otro es la programación central.

La programación de este dispositivo se desarrolló en esta plataforma porque presenta mayor facilidad con las librerías, ya que se pueden enviar en el archivo en el que se trabajó y no necesitan ser añadidas como en el caso de Arduino, ocasionando inconvenientes, además permite el monitoreo de las señales de los sensores de manera más práctica y el lenguaje de programación es el mismo.

3.3.4 Arduino IDE. Debido a los problemas generados por el uso del Wi-Fi es indispensable el uso de esta tarjeta. La arduino pro-mini realiza la recepción y la calibración de datos de los diferentes sensores y los envía en un vector a uno de los pines de la ESP32. Ver archivo APENDICE I. Arduino IDE.

3.3.5 Interfaz. Para realizar la descarga de la batería, el usuario debe ingresar los parámetros con los que desea trabajar por medio de una interfaz gráfica que posee un entorno amigable y de fácil acceso. La interfaz se elaboró en una plataforma gratuita que permite el desarrollo de aplicaciones sin la preocupación de su infraestructura, Heroku es un servicio en la nube tipo *PaaS* plataforma como servicio. Con la ayuda de una API denominada Thingspeak se visualizan y almacenan las señales que permiten estudiar el comportamiento de la carga

electrónica utilizando el principio de internet de las cosas (IoT), se implementó una página web en donde el usuario pueda controlar y decidir sobre cualquier evento que ocurra durante la descarga de la batería.

Para ingresar a la página se debe utilizar la siguiente dirección https://electronic-load-uis.herokuapp.com/ .

Figura 29. Interfaz de carga electrónica

En la figura anterior se puede observar la interfaz gráfica con la que el usuario va a interactuar, en la esquina superior derecha se puede ver el estado de la conexión de la ESP32, así mismo se incluye los dos modos de trabajo con los que se puede descargar la batería: corriente constante y corriente variable, en donde el usuario debe ingresar los parámetros con los que quiere trabajar según sea el caso, además se puede observar el comportamiento de las variables que están siendo monitorizadas en tiempo real en el cuadro superior izquierdo.

Cuando el usuario requiera un estudio más detallado del comportamiento de la carga en un periodo de tiempo puede analizar las diferentes gráficas que se presentan en la página que incluyen, la tensión de la batería, la corriente con la que se descarga la batería, la temperatura del

transistor, la potencia que está entregando la batería y la energía que se refiere a la capacidad de almacenamiento (figura 31).

Figura 30. Comportamiento de las variables de la carga

La página web se desarrolló en visual Studio Code con código HTML, la cual nos permite la unión de los canales seleccionados de la API con una página web, tomando y usando lo necesario de la API en la página web, esto facilita que el usuario realice una monitorización remota del equipo ya sea desde un PC o un móvil.

Para la transmisión de datos de la ESP32 hasta la API se empleó el modo *static* de la esp32, este modo hace necesario que el microcontrolador se conecte a un módem para proceder a subir los datos a la nube y que esta se mantenga conectada para recibir las órdenes necesarias o requeridas.

3.3.6 Prototipo final. En las siguientes figuras se puede observar el prototipo final de la carga electrónica inteligente.

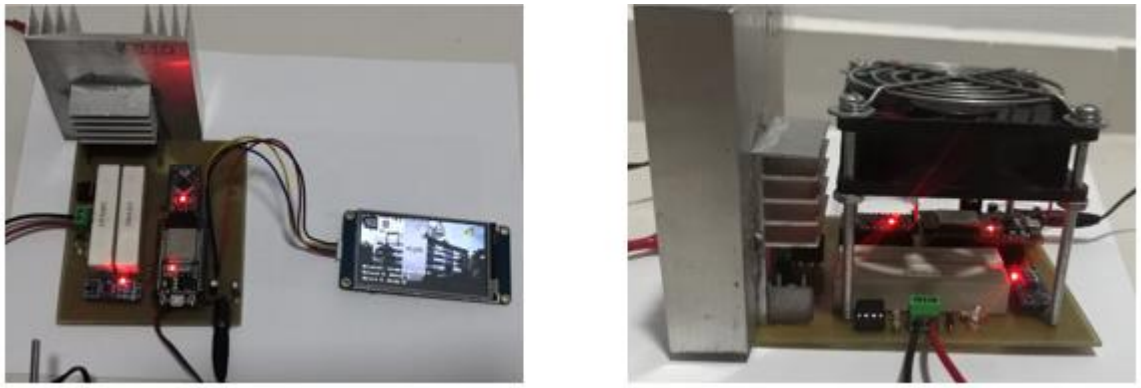


Figura 31. Prototipo final de la carga electrónica

4. Pruebas

Con el fin de verificar el comportamiento de las cargas(una comercial de marca ATORCH y nuestro diseño), se realizaron pruebas con fuentes de 12[V]. Ver tabla 4 y tabla 5.

La primera prueba se realizó con las dos cargas en uno de los laboratorios de la Universidad Industrial de Santander, se tomó una de las fuentes de corriente continua y se utilizó como fuente de tensión de 12 V. Los resultados se ven en la siguiente tabla.

Tabla 4.

Prueba carga Atorch y prototipo

Tensión de la fuente en [V]	Valor teórico de corriente [A]	Corriente de carga Atorch [A]	Corriente del prototipo [A]
12	0	0	0,24
12	0,3	0,35	0,32
12	0,6	0,65	0,61
12	0,9	0,94	0,92
12	1,2	1,22	1,22
12	1,5	1,52	1,5
12	1,8	1,83	1,82
12	2,1	2,13	2,11
12	2,4	2,43	2,43
12	2,7	2,74	2,72
12	3	3,06	2,98

En la prueba que se realizó la tensión en la fuente se mantuvo constante, esto ocurre debido a la que la fuente es de buena calidad y puede entregar la potencia necesaria sin disminuir la tensión. Además, se puede observar que el comportamiento de los dos cargas es similar, sin embargo, el prototipo diseñado presenta valores más cercanos a los teóricos.

Para la carga electrónica diseñada en la UIS, se realizaron pruebas con una batería de un automóvil con tensión de 12[V] y con capacidad de 50[A/h].

Figura 32. Batería de prueba para prototipo

Los resultados se pueden ver en la tabla 5.

Tabla 5.

Pruebas con el prototipo

Tensión de la fuente en [V]	Valor teórico de corriente [A]	Corriente de prototipo [A]	Regulación [%]
12,63	0	0	-
12,55	0,3	0,3	0,02111375
12,43	0,6	0,61	0,025959529
12,27	0,9	0,9	0,031670625
12,15	1,2	1,19	0,031936765
12,07	1,5	1,5	0,02955925
11,95	1,8	1,8	0,029911146
11,83	2,1	2,09	0,030306819
11,6	2,3	2,35	0,034702919
11,59	2,7	2,7	0,030497639
11,51	3	3	0,02955925
11,43	3,3	3,25	0,029234424
11,35	3,6	3,6	0,016891

Con esta prueba se pudo verificar que el estado de la batería es bueno, así mismo se corroboró que el prototipo trabaja de manera adecuada y con datos adecuados para cada uno de los valores asignados.

Con esta misma batería se verificó que cuando la tensión era menor que la tensión de corte (indicada por el fabricante y/o usuario), la descarga terminaba, de igual manera ocurre cuando el usuario ingresa un lapso de tiempo de descarga en la página web y éste es superado por el tiempo de descarga.

Por último, se comprobó el estado de otra batería que también trabaja a 12[V] pero con características diferentes, las cuales se pueden ver en la siguiente imagen

Figura 33. Batería de prueba.

Al momento de realizar la descarga de la batería, se comprobó la tensión sin carga con un valor correspondiente a 12,02[V], sin embargo al momento de exigirle corriente, su tensión decayó a 7,07[V], lo que permite verificar que el estado de la batería es malo y se encuentra dañada.

5. Conclusiones Y Observaciones

- Se seleccionó el transistor MOSFET IRF3205, el cual cumple con los 36 [V] y 4 [A] definidos como límites para la carga electrónica inteligente, en comparación con los BJT. El MOSTEF permitio un funcionamiento mas estable del circuito, es decir que para las mismas variaciones de tensión (V_{GS} y V_{BE}), los cambios de corriente fueron mas suaves en el mosfet que en le BJT.
- Se implementó un circuito de control programado, este control se encarga de manejar las señales analógicas adquiridas, procesarlas y realizar las operaciones aritméticas necesarias para realizar el control PID y que el transistor opere en la zona adecuada para que la corriente con la que se descarga la batería sea la que el usuario desea.
- La corriente con la que se descarga la batería es calculada por el sensor de efecto hall de referencia ACS712, el cual sensa la corriente y arroja un valor de tensión proporcional a ésta, el valor de tensión es leído por un pin analógico del Arduino pro-mini e ingresa a la fórmula $I = \frac{2.5-V}{sensibilidad}$ que lo traduce en corriente por medio del software implementado. El sensor presenta una resistencia de bajo valor (1.2 [mΩ]), lo que quiere decir que sus pérdidas serán insignificantes. La sensibilidad típica del sensor es de 185 mV/A según el fabricante y por pruebas de laboratorio se encontró que esta dentro de las especificaciones con un valor de 150 [mV/A].
- En la pantalla Nextion se desarrolló una interfaz gráfica amigable con el usuario y de fácil acceso, en la cual se puede observar el comportamiento en el lugar en donde se encuentra la carga mediante medidas instantáneas de corriente, tensión, temperatura, potencia y capacidad de la batería. Se desarrolló una página web (aplicativo) junto a un APK donde cada usuario

puede ingresar los parámetros con los que desea descargar la batería y observar las señales que permiten el estudio de su comportamiento de manera remota.

- Se optó por la utilización de dos microcontroladores debido a que la ESP32 presentó problemas de adquisición e interpretación de las señales, generados al iniciar la comunicación con Wi-Fi, obteniendo saltos o en su defecto los valores máximos en la lectura de los sensores afectando el correcto funcionamiento de la carga, por tanto se hizo necesario implementar la tarjeta de desarrollo de arduino pro-mini, que es el dispositivo que permite realizar la adquisición de datos de manera correcta y dividirla de los demás procesos.
- Se elaboró una carga electrónica de 100 [W] por un valor comercial más bajo en comparación a las encontradas y con potencias similares a un precio de $155.000, una diferencia de $70.000 en promedio con respecto a las demás, con la ventaja del monitoreo remoto, del tamaño y además en caso de alguna falla que se presente en la batería el dispositivo móvil estará aislado del dispositivo de control.
- El prototipo de carga electrónica elaborado, maneja valores adecuados de corriente y de tensión, permitiendo realizar la descarga controlada de la batería y calculando de manera adecuada la capacidad de cada una de estas, además, se contrastaron con los valores arrojados por la carga Atorch de 150[W], encontrando que las dos presentan resultados similares en sus lecturas, sin embargo, las diferencias radican en el monitoreo remoto, en que el dispositivo de control está aislado de la carga electrónica evitando que sufra daños frente a posibles fallas, además permite estudiar el comportamiento de las distintas variables de la carga electrónica en el tiempo.
- Se presentaron algunos percances al trabajar con transistores BJT, entré los que se encontró que frente a las variaciones de tensión de descarga de la batería, el rango de trabajo disminuye

si la tensión aumenta y viceversa, generando que el control implementado sea limitado y trabaje correctamente solo para un pequeño rango de tensiones. Otro de los inconvenientes presentados fueron las limitaciones de resolución del circuito, ya que los BJT para su activación necesitan cierta cantidad de corriente en la base que viene dada por un juego de resistencias estático y que permiten que el transistor cambie de estado.

- El prototipo demanda un valor de corriente constante de 0,24[A] que actua como un offset y es independiente de la tensión con la que esté trabajando la batería, esto ocurre debido a que la ESP32 presenta fallas en su ADC, razón por la cual, incluso después de calibradado el ADC sigue presentando corriente parasitas que sólo pueden mejorar su comportamiento con otra tarjeta. Se realizó la prueba y se cambió la ESP32, su resultado fue favorable disminuyendo el valor de la corriente a 0,13[A]. Debido a esto, la carga funciona para descargar baterías con corrientes mayores a 0,4 [A].
- Es indispensable a la hora de descargar cualquier batería, contar con la suficiente precaución de modo que se conecten los bornes de manera correcta: el cable rojo con el terminal positivo de la batería y el cable negro con el terminal negativo de la batería. En el prototipo diseñado se enciende un led verde al conectar la batería de manera correcta, en caso contrario, en el que la batería se conecte inversamente se generaran daños irreversibles.

Referencias bibliografícas

Aliexpress (s.f). Nextion NX4832T035 3,5 pulgadas HMI TFT LCD de pantalla táctil Módulo de 480x320x3,5 "Pantalla táctil resistiva para Raspberry pi 3 Arduino Kit en Accesorios de Tarjeta de demostración de Ordenador y oficina en AliExpress.com | Alibaba Group». [En línea]. Disponible en: https://es.aliexpress.com/item/Nextion-3-5inch-HMI-TFT-LCD-Touch-Display-Module-16M-Flash-480x320-Resistive-Touch-Screen-for/32684571300.html. [Accedido: 19-may-2019].

B. Widlar (1965). Capitulo I-Amplificadores Operacionales Amplificadores Operacionales. Recuperado de: https://www.academia.edu/25099603/CAPITULO_I_AMPLIFICADORES_OPERACIONALES_AMPLIFICADORES_OPERACIONALES

Components 101 (2019). IRF3205 MOSFET Pinout, Datasheet, Features & Alternatives. [En línea]. Disponible en: https://components101.com/mosfets/irf3205-pinout-datasheet. [Accedido: 19-ago-2019].

Direct Industry (s.f). Batería Li-ion / cilíndrica - 3.6 - 3.7 V, 0.68 - 3.25 Ah | NCR - UR series - Panasonic Industry Europe GmbH». [En línea]. Disponible en: http://www.directindustry.es/prod/panasonic-industry-europe-gmbh/product-15208-33754.html. [Accedido: 17-may-2019].

Electroniclab (s.f). Módulo sensor de corriente ACS712 30 A - Electronilab. [En línea]. Disponible en: https://electronilab.co/tienda/modulo-sensor-de-corriente-acs712-30/. [Accedido: 01-jun-2019].

Electronilab (s.f). Board de desarrollo con módulo ESP32 WiFi+Bluetooth BLE - Electronilab». [En línea]. Disponible en: https://electronilab.co/tienda/board-de-desarrollo-con-modulo-esp32-wifibluetooth-ble/. [Accedido: 17-ago-2019].

J. Giles Salazar y Y. Gallego Escalera (2015). Baterías de níquel-hierro (Ni-Fe) by Yaiza Gallego on Prezi. [En línea]. Disponible en: https://prezi.com/uuerjagr9_o9/baterias-de-niquel-hierro-ni-fe/. [Accedido: 16-may-2019].

J. Ramirez Vazquez (1974). Pilas y acomuladores: maquinas de corriente continua, Septima. Ceac, 1974.

J_RPM (2016). Construye una carga electrónica | J_RPM, [En línea]. Disponible en: http://j-rpm.com/2016/04/construye-una-carga-electronica/. [Accedido: 11-may-2019].

JMN electronics (s.f), Re:load. Carga activa. | JMN Electronics. [En línea]. Disponible en: http://jmnelectronics.com/archives/312. [Accedido: 13-may-2019].

L. LLamas (2016). «Controlar grandes cargas con Arduino y transistor MOSFET». [En línea]. Disponible en: https://www.luisllamas.es/arduino-transistor-mosfet/. [Accedido: 14-may-2019].

L. Montero (s.f) El origen de la pila: su evolución y los últimos avances. [En línea]. Disponible en: http://www.proyectofse.mx/2015/07/02/el-origen-de-la-pila/. [Accedido: 02-may-2019].

M. Autor, J. Martínez, y B. Director (2017). Métodos de estimación del estado de carga de baterías electroquímicas Escola Tècnica Superior d'Enginyeria Industrial de Barcelona, Etseib

Mactronica (s.f). LF353. [En línea]. Disponible en: https://www.mactronica.com.co/lf353-144840710xJM. [Accedido: 21-may-2019].

Made In China (s.f). Nife batería de níquel-hierro 1.2V 400Ah para panel solar – Nife batería de níquel-hierro 1.2V 400Ah para panel solar proporcionado por Henan Hengming Fengyun Power Source Co., Ltd. a países hispanohablantes. [En línea]. Disponible en: https://es.made-in-china.com/co_hengmingbattery/product_Nife-Nickel-Iron-Battery-1-2V-400ah-for-Solar-Panel_eyoryyiuy.html. [Accedido: 16-may-2019].

MercadoLibre (s.f). Arduino Pro Mini Atmega328 - $ 55.00 en Mercado Libre». [En línea]. Disponible en: https://articulo.mercadolibre.com.mx/MLM-573623994-arduino-pro-mini-atmega328-_JM?quantity=1. [Accedido: 17-ago-2019].

Naylamp Mechatronics (s.f). Módulo ESP32 ESP-WROOM-32 - Naylamp Mechatronics - Perú». [En línea]. Disponible en: https://naylampmechatronics.com/inalambrico/382-modulo-esp32-esp-wroom-32.html. [Accedido: 18-may-2019].

Naylamp Mechatronics (s.f). Tutorial sensor de corriente ACS712. [En línea]. Disponible en: https://naylampmechatronics.com/blog/48_tutorial-sensor-de-corriente-acs712.html. [Accedido: 01-jun-2019].

O. Gonzalez (2017). Comparativa y análisis completo de los módulos Wifi ESP8266 y ESP32 - BricoGeek.com. [En línea]. Disponible en: https://blog.bricogeek.com/noticias/electronica/comparativa-y-analisis-completo-de-los-modulos-wifi-esp8266-y-esp32/. [Accedido: 18-may-2019].

P. Papageorgas, D. Piromalis, K. Antonakoglou, G. Vokas, D. Tseles, y K. G. Arvanitis (2013). Smart solar panels: In-situ monitoring of photovoltaic panels based on wired and wireless sensor networks, Energy Procedia, vol. 36, n.o December, pp. 535-545P. Typ Max, «IRF3205 HEXFET ® Power MOSFET Absolute Maximum Ratings».

PCFactory (s.f). Adafruit Batería de Polímero de Litio 3.7V 500mAh | PC Factory». [En línea]. Disponible en: https://www.pcfactory.cl/producto/27571-adafruit-bateria-de-polimero-de-litio-3-7v-500mah. [Accedido: 17-may-2019].

Pilasrecargables10 (2018). Pilas recargables NiMH. Baterías de níquel-metal hidruro en 2018». [En línea]. Disponible en: https://www.pilasrecargables10.net/de-niquel-metal-hidruro/. [Accedido: 17-may-2019].

PotentialLabs (s.f). «LM358 – OpAmp – Buy Online in Hyderabad and India». [En línea]. Disponible en: https://potentiallabs.com/cart/lm-358-india. [Accedido: 29-jul-2019].

Random Nerd Tutorials (s.f). Getting Started with the ESP32 Development Board | Random Nerd Tutorials». [En línea]. Disponible en: https://randomnerdtutorials.com/getting-started-with-esp32/. [Accedido: 18-may-2019].

RiverGlennapts (s.f). Transistor como interruptor o unión bipolar transistor o BJT como interruptor». [En línea]. Disponible en: https://riverglennapts.com/transistor/913-transistor-as-a-switch-or-bipolar-junction-transistor-or-bjt-as-switch.html. [Accedido: 29-may-2019].

RS Components (s.f). F734A0566 | Baterías C recargables, Níquel-Cadmio, 3000mAh Lengüetas Terminal, 1.2V | RS Components». [En línea]. Disponible en: https://es.rs-online.com/web/p/pilas-recargables-c/3777703/. [Accedido: 16-may-2019].

RS Components (s.f). TIP3055 | STMicroelectronics | Transistores bipolares y BJT | RS. [En línea]. Disponible en: https://es.rs-online.com/web/p/transistores-bipolares-y-bjt/4859727/. [Accedido: 29-may-2019].

T. Hareendran (2017). Simple Electronic DC Load - Codrey Electronics. [En línea]. Disponible en: https://www.codrey.com/electronic-circuits/simple-electronic-dc-load/. [Accedido: 15-may-2019].

TecnologíaArea (s.f). transistor. [En línea]. Disponible en: https://www.areatecnologia.com/TUTORIALES/EL TRANSISTOR.htm. [Accedido: 14-may-2019].

Texas Instruments (2003). LF353 pdf, LF353 description, LF353 datasheets, LF353 view ::: ALLDATASHEET [En línea]. Disponible en: http://pdf1.alldatasheet.com/datasheet-pdf/view/462264/TI1/LF353.html. [Accedido: 21-may-2019].

Universidad Técnica Federico Santa María (2014). Tecnología de las baterías Definición y Clasificación

V. García (2012). El Transistor MOSFET – Electrónica Práctica Aplicada. [En línea]. Disponible en: https://www.diarioelectronicohoy.com/blog/el-transistor-mosfet. [Accedido: 14-may-2019].

Apendices

Apendice A. Especificaciones de la ESP32.

Tabla 6.

Especificaciones de la ESP32 [30]

Característica	ESP32
Procesador	Tensilica Xtensa LX6 32 bit Dual-Core a 160 MHz (hasta 240 MHz)
Memoria RAM	520 kB
Memoria Flash	Hasta 16 MB
ROM	448 kB
Alimentación	2.2 a 3.6 V
Rango de temperaturas	-40°C a 125°C
Consumo de corriente	80 mA (promedio). 225 mA máximo
Consumo en modo sueño profundo	2.5 uA (10 uA RTC + memoria RTC)
Coprocesador de bajo consumo	Sí. Consumo inferior a 150 uA
WiFi	802.11 b/g/n (hasta +20 dBm) WEP, WPA
Soft-AP	Sí
Bluetooth	v4.2 BR/EDR y BLE
UART	3
I2C	2
SPI	4
GPIO (utilizables)	11
PWM	16
ADC	18 (12 bit)
ADC con preamplificador	Sí (Bajo ruido) Hasta 60 dB
DAC	2 (8 bit)
1-Wire	Implementado por software
I2S	2
CAN bus	1 x 2.0
Ethernet	10/100 Mbps MAC
Sensor de temperatura	Sí
Sensor efecto HALL	Sí
IR	Sí
Temporizadores	4 (64 bits)
Encriptación por hardware	Sí (AES, SHA, RSA, ECC)
Gen. de núm. aleatorios	Sí
Encriptación de la flash	Sí
Arranque seguro	Sí

Se encuentran dos versiones de esta tarjeta, una de 30 pines y otra de 36 pines, para este proyecto se trabajó con la ESP 32 de 30 pines

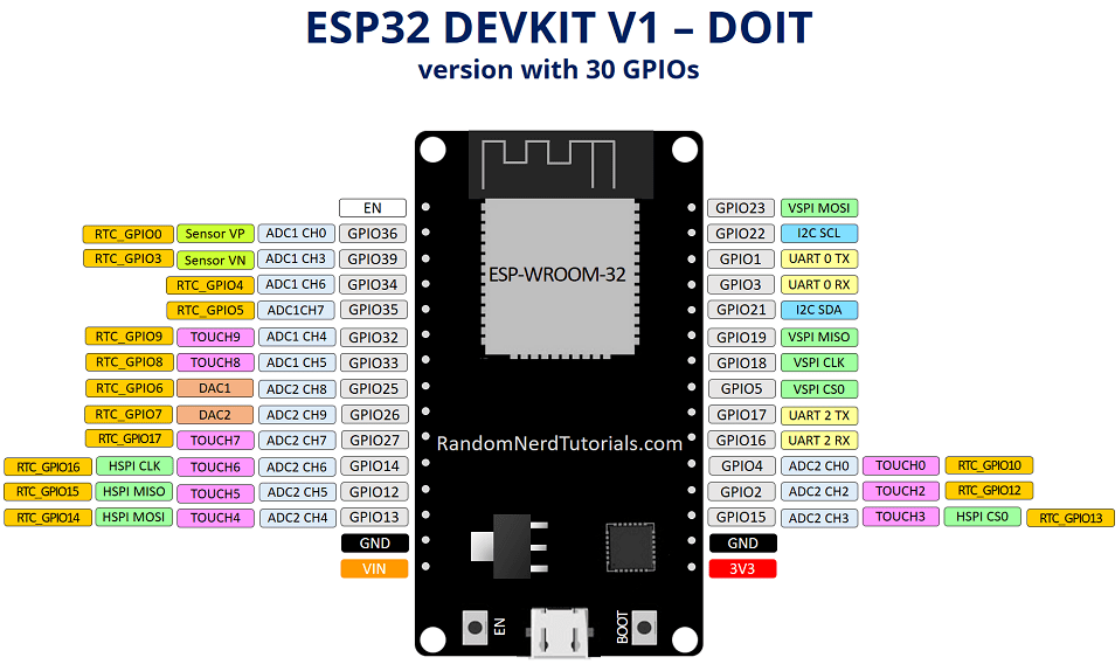


Figura 34. Distribución de pines en la ESP32[31]

Apendice B. Especificaciones De La Nextion

Tabla 7.

Especificaciones Nextion

Especificaciones	
Tarjeta SD	Max de 32 [Gb]
Memoria Flash	16 [MB]
EEPROM	1024 bytes
RAM	3584 bytes
Buffer de instrucciones	1024 bytes
Color	65536 colores
Resolución	400x240 pixeles
Área visual	69.60 [mm] de largo y 41.76 [mm]ancho
Tipo de toque	Resistivo
Luz de fondo	LED
Vida útil retroalimentación	30.000 horas
Fuente de alimentación	5[V] 500 [mA] DC

EEPROM (memoria programable de solo lectura): es una memoria de solo lectura programable y borrable eléctricamente. Es un chip que retiene su contenido sin energía. Funciona como RAM no volátil, pero grabar en EEPROM es mucho más lentor que hacerlo en RAM[23].

Apendice C. Especificaciones del IRF3205

	Parameter	Max.	Units
I_D @ T_C = 25°C	Continuous Drain Current, V_{GS} @ 10V	110 ⑤	A
I_D @ T_C = 100°C	Continuous Drain Current, V_{GS} @ 10V	80	
I_{DM}	Pulsed Drain Current ①	390	
P_D @T_C = 25°C	Power Dissipation	200	W
	Linear Derating Factor	1.3	W/°C
V_{GS}	Gate-to-Source Voltage	± 20	V
I_{AR}	Avalanche Current①	62	A
E_{AR}	Repetitive Avalanche Energy①	20	mJ
dv/dt	Peak Diode Recovery dv/dt ③	5.0	V/ns
T_J T_{STG}	Operating Junction and Storage Temperature Range	-55 to + 175	°C
	Soldering Temperature, for 10 seconds	300 (1.6mm from case)	
	Mounting torque, 6-32 or M3 srew	10 lbf•in (1.1N•m)	

Figura 35. Especificaciones máximas del IRF3205[32]

Apendice D. Especificaciones del LF353

■ Internally trimmed offset voltage:	10 mV
■ Low input bias current:	50pA
■ Low input noise voltage:	25 nV/√Hz
■ Low input noise current:	0.01 pA/√Hz
■ Wide gain bandwidth:	4 MHz
■ High slew rate:	13 V/μs
■ Low supply current:	3.6 mA
■ High input impedance:	$10^{12}\Omega$
■ Low total harmonic distortion :	≤0.02%
■ Low 1/f noise corner:	50 Hz
■ Fast settling time to 0.01%:	2 μs

Supply Voltage	±18V
Power Dissipation	(Note 2)
Operating Temperature Range	0°C to +70°C
T_j(MAX)	150°C
Differential Input Voltage	±30V
Input Voltage Range (Note 3)	±15V
Output Short Circuit Duration	Continuous
Storage Temperature Range	−65°C to +150°C
Lead Temp. (Soldering, 10 sec.)	260°C
Soldering Information	
Dual-In-Line Package	
Soldering (10 sec.)	260°C

Figura 36. Especificaciones del LF353[33].

Apendice E. Especificaciones de TIP3055

Features

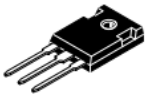
- DC Current Gain –
  h_{FE} = 20–70 @ I_C = 4.0 Adc
- Collector–Emitter Saturation Voltage –
  $V_{CE(sat)}$ = 1.1 Vdc (Max) @ I_C = 4.0 Adc
- Excellent Safe Operating Area
- These are Pb–Free Devices*

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Collector – Emitter Voltage	V_{CEO}	60	Vdc
Collector – Emitter Voltage	V_{CER}	70	Vdc
Collector – Base Voltage	V_{CB}	100	Vdc
Emitter – Base Voltage	V_{EB}	7.0	Vdc
Collector Current – Continuous	I_C	1 5	Adc
Base Current	I_B	7.0	Adc
Total Power Dissipation @ T_C = 25°C Derate above 25°C	P_D	90 0.72	W W/°C
Operating and Storage Junction Temperature Range	T_J, T_{stg}	–65 to +150	°C

15 AMPERE
POWER TRANSISTORS
COMPLEMENTARY SILICON
60 VOLTS, 90 WATTS

SOT–93 (TO–218)
CASE 340D
STYLE 1



TO–247
CASE 340L
STYLE 3

Figura 37. Especificaciones del TIP3055

Apendice F. Características del ACS712

- Voltaje de salida: Analog output 66mV / A
- Voltaje de operación: 4.5V ~ 5.5V
- Salida de voltaje sin corriente: VCC / 2
- Dimensiones PCB: 31 (mm) x14 (mm)
- 5 µs output rise time in response to step input current
- Ancho de banda 80 kHz
- Error Total Salida: 1.5% at TA = 25°C
- Resistencia interna: 1.2 mΩ
- Mínimo voltaje de aislamiento entre pines 1-4 a pines 5-8: 2.1 kVRMS
- Sensitividad de salida: 66 to 185 mV/A

Figura 38. Características del ACS712[34]

Apendice G. Especificaciones LM358

Features

- Available in 8-Bump micro SMD chip sized package, (See AN-1112)
- Internally frequency compensated for unity gain
- Large dc voltage gain: 100 dB
- Wide bandwidth (unity gain): 1 MHz (temperature compensated)
- Wide power supply range:
  — Single supply: 3V to 32V
  — or dual supplies: ±1.5V to ±16V
- Very low supply current drain (500 µA)—essentially independent of supply voltage
- Low input offset voltage: 2 mV
- Input common-mode voltage range includes ground
- Differential input voltage range equal to the power supply voltage
- Large output voltage swing

Figura 39. Especificaciones LM358[35]

Apendice H. Visual Studio Code

Permite la configuración de los principales parámetros de la ESP32, corresponde al archivo config.

```
    #define PIN_DAC 25         //GPIO25 DAC1 - PIN DE SALIDA PARA EL CONTROL DE CORRIENTE
    #define RXD2 13          //Pin para comunicación con el Arduino Pro-mini
    #define TXD2 17          //No se usa pero se define
    #define BAUD_RATE 115200 //Velocidad de comunicación serial

    #define DEVELOPMENT 0
    #define PRODUCTION 1
    // SSID y Contraseña de la primera opción de WiFi
    #define WIFI1 "msvb"
    #define PASSWORD1 "password"
    // SSID y Contraseña de la segunda opción de WiFi
    #define WIFI2 "GERENCIA"
    #define PASSWORD2 "GerenciaPr0C0ntr0l3s2019"

    #define WIFI3 "PROYECTO"
    #define PASSWORD3 "12345678"

    // Configuración de intervalos de muestreo y envio de datos
    #define MAX_OUTPUT 255      //Máximo valor de salida del DAC
    #define OFFSET_TEMP 0.1     //offset de temperatura en grados centígrados
    #define INTERVAL_CONTROL 16 //ms
    #define UPDATE_INTERVAL 700 //Intervalo de envio de datos
    #define EMIT_GUI 500        //microseconds intervalo de control
    #define FINISH_TIME 5000    // millisegundos Tiempo para verificar que termina
    #define MAX_CURRENT 5000    //Máxima corriente en mA
    //#define OUTPUT_PRINT_INTERVAL 1000  //intervalo para ver la salida
    #define ARDUINO_ENV PRODUCTION                      //Modo desarrollo
    #define HOST_SERVER "electronic-load-uis.herokuapp.com" //Servidor socketIO
    #define HOST_DEV "192.168.137.1"                    //Servidor local
    #define PORT_DEV 3000                           //Puerto local
    //
cmd:mode,cutoff[V],(setpoint|initial_current)[A],time_limit[s],step[A],interval[s],current_limit[A]
    #define DEBUG_CONSTANT "constant,3.0,0.4"
    #define DEBUG_VARIABLE "variable,9.5,0.15,60,0.1,10,0.3"
    #define SOCKETIOCLIENT_USE_SSL
```

El archivo main es el que se muestra a continuación.

```
#include <Arduino.h>
#include "config.h"
#include <Nextion.h>
#include <ArduinoJson.h>
#include <WiFi.h>
#include <Ticker.h>
#include <WiFiMulti.h>
#include <WiFiClientSecure.h>
#include <SocketIoClient.h>
#include "utils.cpp"

char arduinoData[64];          // Sensores
byte lastByte1, lastByte2 = 0; // Ultimos 2 bytes de la transmision
uint8_t count = 0;             // Número de bytes
WiFiMulti WiFiMulti;           // Conexión con varias SSID
SocketIoClient io;             // SocketIO client
//Variables para intervalos de tiempo
unsigned long t_start,
    t_start_variable,
    t_start_finish,
    t_start_gui,
    t_start_control;
//measure capacity: [source]: https://www.instructables.com/id/Arduino-cell-capacity-meter/
unsigned long milisec, _time, capacity;
int totalcurrent,         //Corriente total para la suma rieman (integración)
    avgcurrent;           //Promedio de la corriente (Ah)
int vbat,                 //voltaje de bateria
    temp,                 //Temperatura LM35
    current,              //corriente ACS712
    current_limit;        //corriente máxima permitida
unsigned long time_limit; //Tiempo máximo permitido
int current_offset = 0;   //Corriente de offset

int setpoint = 0;                   //Corriente constante deseada
int step = 0;                       //Pasos de aumento de la corriente en modo variable
int cutoff = 0;                     //Tensión de corte
int mode = CONSTANTE;                   // modo
unsigned long interval = 0;           //Intervalo de aumento de corriente
uint8_t run;                        //Estado del control 1=iniciado 0=detenido 2=pausado
uint8_t out = 0, lastOut = 1, base_out; //out DAC
//Variables para GUI y display
float _vbat, _current, _temp, _capacity;
```

```
//Variables de nextion
NexText tVbat = NexText(1, 8, "vbat");
NexText tCurrent = NexText(1, 9, "current");
NexText tPower = NexText(1, 10, "power");
NexText tTemp = NexText(1, 11, "temp");
NexText tCapacity = NexText(1, 12, "capacity");
NexText tTime = NexText(1, 13, "time");
//Ticker para parpadear el LED si no hay conexión WiFi
Ticker blinker;
String payloadArduino;
void blink()
{
  digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
}
void toggleWifi(bool status)
{
  static bool isBlinking = false;
  if (isBlinking != status)
  {
    if (isBlinking)
    {
      blinker.detach();
      isBlinking = false;
    }
    else
    {
      blinker.attach(1.0, blink);
      isBlinking = true;
    }
    digitalWrite(LED_BUILTIN, LOW); //make sure LED on on after toggling (pin LOW = led ON)
  }
}
void clearArduinoData()
{ //Limpia los datos recibidos
  Serial1.flush();
  if (count > 0)
    memset(arduinoData, 0, count);
  count = 0;
}
void changePage(uint8_t num, uint8_t retries = 2)
{
  const char *cmd = String("page " + String(num)).c_str();
  bool ret = false;
  uint8_t c = 0;
  while (!ret)
```

```
  {
    sendCommand(cmd);
    ret = recvRetCommandFinished();
    if (++c >= retries)
      break;
  }
}
// Finaliza el control
void finish(const char *status)
{
  setpoint = 0;
  cutoff = 0;
  _time = 0;
  mode = CONSTANTE;
  step = 0;
  milisec = 0;
  interval = 0;
  totalcurrent = 0;
  capacity = 0;
  out = 0;
  lastOut = 1;
  setOutput(0);
  run = 0;
  t_start_finish = 0;
  if (io.connected)
  {
    io.emit("run", "0");
    io.emit("finish", status);
  }
  changePage(0);
  Serial.println("[FINISH]");
}
//Evento cuando se conecta el socket
void onConnect(const char *payload, size_t length)
{
  if (io.connected)
    io.emit("run", String(run).c_str());
}

void onDisconnect(const char *payload, size_t length)
{
}
// Evento para iniciar el control
void eventStart(const char *payload, size_t length)
{
  //Los valores vienen en un array, es un string separados por comas
```

```
char *command = strtok((char *)payload, ",");
int i = 0;
while (command != 0)
{
  switch (i)
  {
  case 0:
    mode = String(command) == "constant" ? CONSTANTE : VARIABLE;
    break;
  case 1:
    cutoff = (int)(atof(command) * 1000);
  case 2:
    setpoint = (int)(atof(command) * 1000);
    break;
  case 3:
    time_limit = (unsigned long)(atof(command) * 1000);
    break;
  case 4:
    step = (int)(atof(command) * 1000);
    break;
  case 5:
    interval = (unsigned long)(atof(command) * 1000);
    break;
  case 6:
    current_limit = (int)(atof(command) * 1000);
    break;
  }
  i++;
  command = strtok(0, ",");
}
if (mode == CONSTANTE)
{
  interval = 0;
  step = 0;
  current_limit = 0;
}
else
{
  if (current_limit == 0)
    current_limit = 4000;
}
_time = 0;
avgcurrent = 0;
capacity = 0;
totalcurrent = 0;
_current = float(setpoint / 1000.0);
```

```
      base_out = int(setpoint * 0.038f);
      out = base_out;
      lastOut = 1;
      t_start_finish = 0;
      if (io.connected == true)
        io.emit("run", "1");
      changePage(1);
      Serial.printf("[START] %s cutoff:%.2f setpoint:%.2f step:%.2f interval:%.2f out:%d\r\n",
mode == CONSTANTE ? "constant" : "variable", (cutoff / 1000.0), (setpoint / 1000.0), step > 0
? (step / 1000.0) : 0, interval > 0 ? (interval / 1000.0) : 0, out);
      setOutput(out);
      delay(1000);
      milisec = millis();
      run = 1;
    }

    void eventStop(const char *payload, size_t length)
    {
      finish("0");
      Serial.println("[STOP]");
    }

    void eventPause(const char *payload, size_t length)
    {
      out = 0;
      setOutput(0);
      if (io.connected)
        io.emit("run", "2");
      Serial.println("[PAUSE]");
      run = 2;
    }

    void eventResume(const char *payload, size_t length)
    {
      if (io.connected)
        io.emit("run", "1");
      Serial.println("[RESUME]");
      milisec = millis();
      run = 1;
    }

    void setup()
    {
      Serial.begin(250000);
      pinMode(LED_BUILTIN, OUTPUT);
      setOutput(0);
```

```
#if ARDUINO_ENV == DEVELOPMENT
  Serial.printf("\r\nServidor SocketIO:%s\r\n", HOST_DEV);
#else
  Serial.printf("\r\nServidor SocketIO:%s\r\n", HOST_SERVER);
#endif
  for (uint8_t t = 4; t > 0; t--)
  {
    Serial.printf("[SETUP] BOOT WAIT %d...\n", t);
    Serial.flush();
    delay(1000);
  }
  Serial1.begin(BAUD_RATE, SERIAL_8N1, RXD2, TXD2);
  Serial.flush();
  Serial1.flush();
  //Añade las posibles redes que se puede conectar
  WiFiMulti.addAP(WIFI3, PASSWORD3);
  WiFiMulti.addAP(WIFI1, PASSWORD1);
  WiFiMulti.addAP(WIFI2, PASSWORD2);

  Serial.println("Realizado conexión WiFi");
  //Realiza la conexión WiFi
  while (WiFiMulti.run() != WL_CONNECTED)
  {
    blink();
    delay(100);
  }
  Serial.println("Configurando SocketIO");
  io.on("connect", onConnect);
  io.on("disconnect", onDisconnect);
  io.on("start", eventStart);
  io.on("stop", eventStop);
  io.on("pause", eventPause);
  io.on("resume", eventResume);
#if ARDUINO_ENV == DEVELOPMENT
  io.begin(HOST_DEV, PORT_DEV);
#else
  io.beginSSL(HOST_SERVER);
#endif
  Serial.println("Iniciando Carga Electronica Inteligente");
  nexInit();
  setOutput(0);
}

void loop()
{
  //LED indicador del estado del WiFi
```

```
    toggleWifi(WiFi.status() != WL_CONNECTED);
    /**
     * Ejecuta un test rapido, ingresando el comando `const` o `var` en el monitor serial
     * */
    if (Serial.available())
    {
      String data = Serial.readStringUntil('\n');
      if (data.indexOf("const") > -1)
      {
        const char payload[] = DEBUG_CONSTANT;
        eventStart(payload, strlen(payload));
      }
      if (data.indexOf("var") > -1)
      {
        const char payload[] = DEBUG_VARIABLE;
        eventStart(payload, strlen(payload));
      }
      else if (data.indexOf("pause") > -1)
      {
        eventPause(nullptr, 0);
      }
      else if (data.indexOf("resume") > -1)
      {
        eventResume(nullptr, 0);
      }
      else if (data.indexOf("stop") > -1)
      {
        eventStop(nullptr, 0);
      }
    }
    /**
     * Read serial data sensors in 6 bytes + 3 bytes to ACK from arduino
     * */
    if (Serial1.available())
    {
      payloadArduino = Serial1.readStringUntil('>');
      //"<0,2546,0>"
      if (payloadArduino.startsWith("<") && payloadArduino.length() >= 9)
      {
        payloadArduino = payloadArduino.substring(1, payloadArduino.length());
        if (payloadArduino.indexOf("<") == -1)
        {
          char *command = strtok((char *)payloadArduino.c_str(), ",");
          byte i = 0;
          while (command != 0)
          {
```

```
          switch (i)
          {
          case 0:
            vbat = (int)atoi(command);
            break;
          case 1:
            temp = (int)atoi(command) - (OFFSET_TEMP * 100);
            break;
          case 2:
            current = (int)atoi(command);
            break;
          }
          i++;
          command = strtok(0, ",");
        }
        //Valores X10^0
        _temp = float(temp / 100.0);
        _vbat = float(vbat / 1000.0);
        _current = float(current / 1000.0);
      }
    }
  }
  /**COMPUTE OUTPUT */
  if (millis() - t_start_control >= INTERVAL_CONTROL)
  {
    if (run == 1) // Verifica si el control fue iniciado
    {
      if (mode == VARIABLE && interval > 0 && step > 0) //modo variable
      {
        if (millis() - t_start_variable >= interval)
        {
          setpoint += step;
          base_out = int(setpoint * 0.038f);
          out = base_out;
          setOutput(out);
          Serial.printf("Interval update: %.2f\r\n", float(setpoint / 1000.0));
          t_start_variable = millis();
        }
      }
      _time = (millis() - milisec); // compute time
      if (_time > 0)
      {
        totalcurrent += current; // calculate total amps
        if (totalcurrent > 0)
        {
          avgcurrent = totalcurrent / _time;      // average amps
```

```
      capacity = (avgcurrent * _time) / 3600; // amp-hour
      _capacity = capacity / 1000.0;
    }
  }
  if (out <= 5 && mode == CONSTANTE)
    out = base_out;

  if (current < setpoint)
  {
    if (++out > MAX_OUTPUT)
      out = MAX_OUTPUT;
  }
  else if (current > setpoint)
  {
    if (--out < 0)
    {
      out = 0;
    }
  }
  if (current <= MAX_CURRENT) // verifica la corriente máxima
  {                           //Verifica que no se sobrepase de la corriente maxima
    setOutput(out);
    lastOut = out;
  }

  if (vbat < cutoff)
  {
    if (t_start_finish == 0)
      t_start_finish = millis();
    else if (millis() - t_start_finish >= FINISH_TIME)
    {
      t_start_finish = 0;
      finish("1");
      Serial.printf("[CUTOFF] vbat:%.2fV\r\n", _vbat);
    }
  }
  if (_time > 1000 && time_limit > 0 && _time >= time_limit)
  {
    finish("2");
    Serial.printf("[LIMIT_TIME] time:%.2fs\r\n", float(_time / 1000.0));
  }
  if ((current_limit > 0 && current > current_limit) || (current > MAX_CURRENT))
  {
    finish("3");
    Serial.printf("[MAX CURRENT LIMIT] vbat:%.2fV %.2fA\r\n", _vbat, _current);
  }
```

```
      }
      t_start_control = millis();
    }
    //Send data to socket
    if (millis() - t_start_gui > EMIT_GUI)
    {
      if (io.connected == true)
      {
        char payload[64];
        unsigned long t = run == 1 ? float((millis() - milisec) / 1000.0) : 0;
        sprintf(payload, "%.2f,%.2f,%.2f,%.2f,%ld", _vbat, _temp, run == 1 ? _current : 0,
_capacity, t);
        io.emit("data", payload);
      }
      t_start_gui = millis();
    }
    //Send data to Display Nextion and Debug Serial
    if (millis() - t_start >= UPDATE_INTERVAL)
    {
      char data[12];
      if (run == 1)
      {
        //Print vbat in nextion
        memset(data, 0, sizeof(data));
        sprintf(data, "%.2fV", _vbat);
        tVbat.setText(data);
        //Print current in nextion
        memset(data, 0, sizeof(data));
        sprintf(data, "%.2fA", _current);
        tCurrent.setText(data);
        //Print power in nextion
        memset(data, 0, sizeof(data));
        sprintf(data, "%.2fW", float(_vbat * _current));
        tPower.setText(data);
        //Print temp in nextion
        memset(data, 0, sizeof(data));
        sprintf(data, "%.2fC", _temp);
        tTemp.setText(data);
        //Print capacity in nextion
        memset(data, 0, sizeof(data));
        sprintf(data, "%.2fAh", _capacity);
        tCapacity.setText(data);
      }
      unsigned long t = run == 1 ? float((millis() - milisec) / 1000.0) : 0;
      int hours = numberOfHours(t);
      int minutes = numberOfMinutes(t);
```

```
        int seconds = numberOfSeconds(t);
        //Print Time in nextion
        memset(data, 0, sizeof(data));
        sprintf(data, "%02d:%02d:%02d", hours, minutes, seconds);
        if (run == 1)
          tTime.setText(data);
        float _setpoint = setpoint > 0 ? (float)setpoint / 1000.0 : 0;
        float _cutoff = cutoff > 0 ? (float)cutoff / 1000.0 : 0;
        float _step = step > 0 ? (float)step / 1000.0 : 0;
        float _interval = interval > 0 ? (float)interval / 1000.0 : 0;
        float _time_limit = time_limit > 0 ? (float)time_limit / 1000.0 : 0;
        float _current_limit = current_limit > 0 ? (float)current_limit / 1000.0 : 0;
        //Print data

Serial.printf("vbat:%.2fV\t%.2fC\tcurrent:%.2fA\t%s\tsetpoint:%.2fA\tcutoff:%.2fv\t%.2fA/h\tst
ep:%.2fA\tt:%.2fs\tmax:%2.fs\tmax:%.2fA\tout:%u\r\n", _vbat, _temp, _current, data, _setpoint,
_cutoff, _capacity, _step, _interval, _time_limit, _current_limit, out);
        t_start = millis();
      }
      if (run != 1)
        setOutput(0);
      io.loop();
    }
```

Apendice I. Arduino IDE

```
#define PIN_LM35 A1
#define PIN_BATTERY A2
#define PIN_ACS712 A0
#define COEF_DIV 8.15f//Coeficiente de divisor v = input voltage ÷ output voltage
#define PIN_3V3 A6 //Pin para voltaje de referencia 3.3V
#define DEBUG 0 //HABILITA LA DEPURACION
//467M
//97K
#define NUM_SAMPLES 10
#define ADC_SCALE 1023.0
#define VREF 5.0
#define DEFAULT_FREQUENCY 50

class SmoothAnalog {
  public:
    SmoothAnalog(int pin) {
      _pin = pin;
      // initialize all the readings to 0:
      for (int thisReading = 0; thisReading < NUM_SAMPLES; thisReading++)
        readings[thisReading] = 0;
    }
    //Funcion smooth Arduino
    int read() {
      total -= readings[readIndex];
      readings[readIndex] = analogRead(_pin);
      total += readings[readIndex];
      if (++readIndex >= NUM_SAMPLES)
        readIndex = 0;
      average = (total / NUM_SAMPLES);
      return average;
    }
    int average = 0;
  private:
    int _pin;
    int readings[NUM_SAMPLES];     //Lecturas
    int readIndex = 0;   //indice de lecturas
    long total = 0;       //Suma total
};
SmoothAnalog vref = SmoothAnalog(PIN_3V3);
//float getVoltsFix(int rawValue) {
//  int value = vref.read();
//  //if (vref3v3 < 500 || vref3v3 > 750) return float(float(rawValue * VREF) / ADC_SCALE);
//  Serial.println(rawValue / value, 5);
```

```
//  Serial.println(rawValue );
//  Serial.println(value);
//  return roundf(float(rawValue / value) * 3.30);
//}
class LM35 : public SmoothAnalog {
  public:
    LM35(int pin) : SmoothAnalog(pin) {};
    int getTemp(int shift = 1) {
      return round(((read() * VREF * 100.0) / (ADC_SCALE / shift)));
    }
};
enum ACS712_type {ACS712_05B, ACS712_20A, ACS712_30A};
class ACS712 : public SmoothAnalog {
  public:
    ACS712(ACS712_type type, int _pin) : SmoothAnalog(_pin) {
      switch (type) {
        case ACS712_05B:
          sensitivity = 0.185;
          break;
        case ACS712_20A:
          sensitivity = 0.100;
          break;
        case ACS712_30A:
          sensitivity = 0.066;
          break;
        default:
          sensitivity = 0.066;
          break;
      }
      pin = _pin;
    }
    int calibrate() {
      for (int i = 0; i < 100; i++) {
        zero = read();
        delay(1);
      }
      return zero;
    }
    void setZeroPoint(int _zero) {
      zero = _zero;
    }
    void setSensitivity(float sens) {
      sensitivity = sens;
    }
    float getCurrentDC() {
      int value = read();
```

```
    if ((value - zero) <= 1) return 0.0f;
    float I = float(float((value - zero) * VREF) / ADC_SCALE) / sensitivity;
    return I;
   }

   float getCurrentAC() {
    return getCurrentAC(DEFAULT_FREQUENCY);
   }

   int getMilliAmpers() {
    return getCurrentDC() * 1000.0;
   }

   float getCurrentAC(uint16_t frequency) {
    uint32_t period = 1000000 / frequency;
    uint32_t t_start = micros();

    uint32_t Isum = 0, measurements_count = 0;
    int32_t Inow;

    while (micros() - t_start < period) {
     Inow = zero - read();
     Isum += Inow * Inow;
     measurements_count++;
    }

    float Irms = sqrt(Isum / measurements_count) / ADC_SCALE * VREF / sensitivity;
    return Irms;
   }
  private:
   float zero = 512.0;
   float sensitivity;
   uint8_t pin;
};

class Battery : public SmoothAnalog {
  public:
   Battery(int pin) : SmoothAnalog(pin) {
   };

   float getVoltage() {
    return ((read() * VREF) / ADC_SCALE) * COEF_DIV;
   }
   int getMilliVolts()
   {
    return getVoltage() * 1000.0;
```

```
    }
};

LM35 lm35 = LM35(PIN_LM35);
ACS712 acs712 = ACS712(ACS712_05B, PIN_ACS712);
Battery battery = Battery(PIN_BATTERY);
int vbat, temp, current, current2;

void setup() {
  delay(1000);
  int zero = acs712.calibrate();  // initialize serial communication with computer:
  Serial.begin(115200);
  //Serial.println("Calibrando ACS712...");
  //Serial.println("Zero: " + String(zero));
}

void loop() {
  vbat = battery.getMilliVolts();
  delay(1);       // delay in between reads for stability
  temp = lm35.getTemp(100);
  delay(1);       // delay in between reads for stability
  current = acs712.getMilliAmpers();
  if (current < 64) current = 0;
  if (vbat < 30) vbat = 0;
  sendESP();

}

void sendESP() {
  char data[64];
  sprintf(data, "<%d,%d,%d>", (int)vbat, (int)temp, (int)current);
  Serial.print(data);

  //  sendData(vbat);
  //  sendData(temp);
  //  sendData(current);
  //  Serial.write(0xFF);
  //  Serial.write(0xFF);
  //  Serial.write(0xFF);
}

void sendData(int data) {
  Serial.write(data >> 8);
  Serial.write(data & 0xFF);
}
```

```
void debug() {
  Serial.print("vbat:");
  Serial.print(vbat);
  Serial.print("  ");
  Serial.print("temp:");
  Serial.print(temp);
  Serial.print("  ");
  Serial.print("current:");
  Serial.print(current);
  Serial.println();
}
```